

JEDI

52

JUILLET 1989

LE JOURNAL QUI NE REVEILLE PAS LES MONSTRES



TURBO-FORTH
A HAUTE
DENSITE:

.SS, POUR
VISU PILE

BANNIERE
ECRAN

ON..GOSUB
TURBO

VISU IMAGES
DIGITALISEES

TOURS DE
HANOI

DESASSEM-
BLEUR 8086

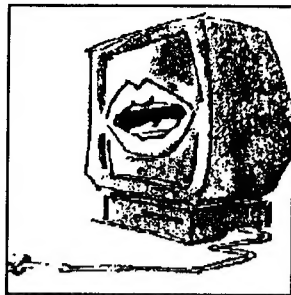
ETC....



EDITORIAL

Alors comme ça, j'ai appris que l'on voulait réformer l'orthographe? Que notre langue soit compliquée, certes, je l'admet. Mais peut-on faire table rase de tous les éléments linguistiques marquant notre langue? Non! Si les enfants ont des difficultés en orthographe, il faut peut-être en rechercher la cause ailleurs qu'en argumentant sur la seule complexité grammaticale. Parmi les enfants (et les autres) se gavant d'émissions serinant à longueur de journée des japonaiseries insipides, des jeux ressemblant de plus en plus à de très longs spots publicitaires, combien lisent-ils de livres, de journaux? Pas beaucoup, je pense... L'argument comme quoi les distinctions sociales seront atténuées dès que le principe d'une grammaire simplifiée sera admis est inconsistant. Tenez, je vous écris la prochaine phase en grammaire simplifiée. Si un tel poje de simplifikasion de l'ortograp abouti, nou riskon de voua aparètr une nouvèl form de discriminasion entr seu écrivain comm lé ansien et lé modern masakran lé règl; j'arrête le massacre!

Dans le cas des homophones, ils ne peuvent souvent être identifiés, indépendamment de leur contexte,



que grâce à leur orthographe.

Ceux-là même prêchant cette simplification, ont-ils vérifié l'efficacité des méthodes pédagogiques actuellement en place. Peut-être est-ce de ce côté qu'il faudrait faire un effort. Prenons le cas du plan INFORMATIQUE POUR TOUS, destiné à équiper toutes les écoles en micro-ordinateurs; ces machines devaient permettre aux élèves de se servir d'outils conviviaux et pédagogiques pour faciliter leur apprentissage dans de nombreuses disciplines, dont l'orthographe. Si la démarche est louable, son efficacité semble complètement compromise. En effet, voici un secteur où l'alliance CD-ROM et micro-ordinateur pouvait faire des merveilles. Dans la pratique, on a surtout doré la pillule à Mr THOMSON qui a pu fourguer son parc de machines devenu complètement obsolète le temps que les utilisateurs le prennent en main.

Personnellement, je ne crois pas à l'efficacité d'un décret instaurant de nouvelles règles grammaticales. Les règles et le langage évoluent indépendamment de notre volonté. Ceci est affaire de culture et l'influence des médias et la technologie nous modèlent davantage que les décisions arbitraires d'un petit comité. Si on déplore la pratique du franglais (intégration de mots d'origine anglo-saxonne), consolons-nous en constatant que ce phénomène est général: russ-glais, ital-français...

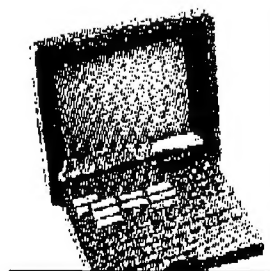
SOMMAIRE

FORTH: Paramètres et pointeurs de pile	2
Bannière écran	2
Structure OUT-ENDOUT	4
Visualisation d'images digitalisées	5
Turbo-Forth résident	7
Tours de Hanoi	9
Désassembleur 8086 pour TF83	12
Création de listes de variables et de constantes	16
TELEMATIQUE: Mode d'emploi du réseau BTX	3
Contenu du Forum SAM*JEDI	17

Toute reproduction, adaptation, traduction partielle du contenu de ce magazine sous toutes les formes est vivement encouragée, à l'exception de toute reproduction à des fins commerciales. Dans le cas de reproduction par photocopie, il est demandé de ne pas masquer les références inscrites en bas de page, et dans les autres cas, de citer l'ASSOCIATION JEDI.

Nos coordonnées: ASSOCIATION JEDI
17, rue de la Lancette
75012 PARIS
tél président: (33) 1-43.40.96.53
tél secrétaire: (33) 1-49.85.63.67

Télétel:
3615 SAM*JEDI (France, DOM, TOM)
(33) 36.43.15.15 (1200E71) (Etranger)



FORTH

PARAMETRES ET POINTEURS DE PILE

par Bill Beers
adaptation Marc PETREMANN

Adaptable: F83 Laxen et Perry CP/M et MS-DOS,
VolksForth ATARI.
Diffusion: 3615 SAM*JEDI et module 7 TURBO-Forth

LISTING: DOTS.FTH

ECHO OFF

\ A more informative stack display. By Bill Beers.
\ Extracted for the East Coast Forth BBS by Steve Palincsar.

\ For Laxen & Perry F83 for IBM PC and compatibles.
\ Adapted to TURBO-Forth by Marc PETREMANN,
\ ASSOCIATION JEDI (FRANCE)
\ Adapté à TURBO-Forth par Marc PETREMANN
\ Diffusion by ASSOCIATION JEDI (diffusé par)
\ 17, rue de la Lancette
\ F-75012 PARIS - France
\ Phone (tél): (33) 1-43409653
\ mailbox (serveur): 3615 SAM*JEDI
\ (1200/75 bauds E71) from FRANCE only
\ (33) 36.43.15.15 SAM*JEDI for foreigner
\ --- 32 simultaneous access. ---

INCLUDE POLYGLOT \ load the multi-lingual manager
\ charge le gestionnaire poly-linguistique

ENGLISH ?\ ECHO ON

\ The user word is ".SS". Use instead of '.S'.
\ Gives you a display (in vertical format) of what is on the
\ stack in Decimal, Binary and Hex notation, and the
\ addresses of the top and bottom of the stack.

ECHO OFF

FRENCH ?\ ECHO ON

\ Le mot clé utilisé est ".SS". Utilisé comme '.S'.
\ Affiche (en format vertical) le contenu de la pile dans
\ les bases numériques décimales, hexadécimales et binaires,
\ ainsi que les adresses du sommet et la base de la pile.

ECHO OFF

```
: BINARY 2 BASE ! ; HEX
: FMT-BIT # # # # 2D HOLD # # # # BL HOLD ;
: BITS S>D <# FMT-BIT #> TYPE ;
: .CNT ( n --- ) DECIMAL 3 .R ." |" ;
: TOPSTK ( ---)
  CR
  ENGLISH ?\ ." ——— Top of Stack : " SP@ 2+ U.
  ENGLISH ?\ ." ————"
  FRENCH ?\ ." ——— Sommet de la pile : " SP@ 2+ U.
  FRENCH ?\ ." ————"
  GERMAN ?\ ." ——— Stack oben : " SP@ 2+ U.
  GERMAN ?\ ." ————"
  CR ;
: BOTSTK ( ---)
  ENGLISH ?\ ." ——— Bottom of Stack : " SP0 @ U.
  ENGLISH ?\ ." ————"
  FRENCH ?\ ." ——— Bas de la pile : " SP0 @ U.
  FRENCH ?\ ." ————"
  GERMAN ?\ ." ——— Stack unter : " SP0 @ U.
  GERMAN ?\ ." ————"
  CR ;
: <.S> ( n --- ) >R R@ DECIMAL 5 U.R ." dec|"
  hex R@ 5 U.R ." hex|" octal R@ 7 u.r ." oct|"
  BINARY R> BITS ;
: .SS BASE @ R DEPTH ?DUP
```

```
  i pick <.s> cr key? ?leave
  loop
  R> BASE ! BOTSTK
  ELSE R> BASE !
  ENGLISH ?\ CR ." <<< Stack Empty >>>"
  FRENCH ?\ CR ." <<< Pile Vide >>>"
  GERMAN ?\ CR ." <<< Leere Stack >>>"
  THEN ;
decimal
```

FORTH

BANNIERE ECRAN

Par Tom ZIMMER
adaptation Marc PETREMANN

Adaptable: F83 Laxen et Perry CP/M et MS-DOS,
VolksForth ATARI.
Diffusion: 3615 SAM*JEDI et module 7 TURBO-Forth

LISTING: BANNER.FTH

ECHO OFF

\ BANNER.SEQ Compliments of F83X
\ mod to sequential by Tom Zimmer
\ Adapted for TURBO-Forth by Marc PETREMANN
\ Adapté à TURBO-Forth par Marc PETREMANN
\ Diffusion by ASSOCIATION JEDI (diffusé par)
\ 17, rue de la Lancette
\ F-75012 PARIS - France
\ tel: (33) 1-43409653
\ mailbox (serveur): 3615 SAM*JEDI
\ (1200/75 bauds E71) from FRANCE
\ (33) 36.43.15.15 SAM*JEDI for foreigner

INCLUDE POLYGLOT \ Multi-lingual module for TURBO-Forth
\ Chargement utilitaire multi-langue
CREATE CHAR-MATRIX \ build the character generator
\ construit le générateur de caractères

```
HEX ( ) 00 C, 00 C, 00 C, 00 C, 00 C, 00 C, 00 C, 00 C,
( 1) 20 C, 20 C, 20 C, 20 C, 20 C, 00 C, 20 C, 00 C,
( 2) 50 C, 50 C, 50 C, 00 C, 00 C, 00 C, 00 C, 00 C,
( 3) 50 C, 50 C, F8 C, 50 C, F8 C, 50 C, 50 C, 00 C,
( 4) 20 C, 78 C, A0 C, 70 C, 28 C, F0 C, 20 C, 00 C,
( 5) C0 C, C8 C, 10 C, 20 C, 40 C, 98 C, 18 C, 00 C,
( 6) 40 C, A0 C, A0 C, 40 C, A8 C, 90 C, 68 C, 00 C,
( 7) 30 C, 30 C, 10 C, 20 C, 00 C, 00 C, 00 C, 00 C,
( 8) 20 C, 40 C, 80 C, 80 C, 80 C, 40 C, 20 C, 00 C,
( 9) 20 C, 10 C, 08 C, 08 C, 08 C, 10 C, 20 C, 00 C,
( A) 20 C, a8 C, 70 C, 20 C, 70 C, a8 C, 20 C, 00 C,
( B) 00 C, 20 C, 20 C, 70 C, 20 C, 20 C, 00 C, 00 C,
( C) 00 C, 00 C, 00 C, 30 C, 30 C, 10 C, 20 C, 00 C,
( D) 00 C, 00 C, 00 C, 70 C, 00 C, 00 C, 00 C, 00 C,
( E) 00 C, 00 C, 00 C, 00 C, 00 C, 30 C, 30 C, 00 C,
( F) 00 C, 08 C, 10 C, 20 C, 40 C, 80 C, 00 C, 00 C,
( 0) 70 C, 88 C, 98 C, A8 C, C8 C, 88 C, 70 C, 00 C,
( 1) 20 C, 60 C, 20 C, 20 C, 20 C, 20 C, 70 C, 00 C,
( 2) 70 C, 88 C, 08 C, 30 C, 40 C, 80 C, F8 C, 00 C,
( 3) F8 C, 10 C, 20 C, 30 C, 08 C, 88 C, 70 C, 00 C,
( 4) 10 C, 30 C, 50 C, 90 C, F8 C, 10 C, 10 C, 00 C,
( 5) F8 C, 80 C, F0 C, 08 C, 08 C, 88 C, 70 C, 00 C,
( 6) 38 C, 40 C, 80 C, F0 C, 88 C, 88 C, 70 C, 00 C,
( 7) F8 C, 08 C, 10 C, 20 C, 40 C, 40 C, 40 C, 00 C,
( 8) 70 C, 88 C, 88 C, 70 C, 88 C, 88 C, 70 C, 00 C,
( 9) 70 C, 88 C, 88 C, 78 C, 08 C, 10 C, E0 C, 00 C,
( : ) 00 C, 60 C, 60 C, 00 C, 60 C, 60 C, 00 C, 00 C,
( ; ) 00 C, 60 C, 60 C, 00 C, 60 C, 60 C, 40 C, 00 C,
( < ) 10 C, 20 C, 40 C, 80 C, 40 C, 20 C, 10 C, 00 C,
( = ) 00 C, 00 C, F8 C, 00 C, F8 C, 00 C, 00 C, 00 C,
```

```

( @ ) 70 C, 88 C, A8 C, B8 C, B0 C, 80 C, 78 C, 00 C,
( A ) 20 C, 70 C, 88 C, 88 C, F8 C, 88 C, 88 C, 00 C,
( B ) F0 C, 88 C, 88 C, F0 C, 88 C, 88 C, F0 C, 00 C,
( C ) 70 C, 88 C, 80 C, 80 C, 80 C, 88 C, 70 C, 00 C,
( D ) F0 C, 48 C, 48 C, 48 C, 48 C, 48 C, F0 C, 00 C,
( E ) F8 C, 80 C, 80 C, F0 C, 80 C, 80 C, F8 C, 00 C,
( F ) F8 C, 80 C, 80 C, F0 C, 80 C, 80 C, 80 C, 00 C,
( G ) 78 C, 80 C, 80 C, 80 C, 98 C, 88 C, 78 C, 00 C,
( H ) 88 C, 88 C, 88 C, F8 C, 88 C, 88 C, 88 C, 00 C,
( I ) 70 C, 20 C, 20 C, 20 C, 20 C, 20 C, 70 C, 00 C,
( J ) 08 C, 08 C, 08 C, 08 C, 08 C, 08 C, 78 C, 00 C,
( K ) 88 C, 90 C, A0 C, C0 C, A0 C, 90 C, 88 C, 00 C,
( L ) 80 C, 80 C, 80 C, 80 C, 80 C, 80 C, F8 C, 00 C,
( M ) 88 C, D8 C, A8 C, A8 C, 88 C, 88 C, 88 C, 00 C,
( N ) 88 C, 88 C, C8 C, A8 C, 98 C, 88 C, 88 C, 00 C,
( O ) 70 C, 88 C, 88 C, 88 C, 88 C, 88 C, 70 C, 00 C,
( P ) F0 C, 88 C, 88 C, F0 C, 80 C, 80 C, 80 C, 00 C,
( Q ) 70 C, 88 C, 88 C, 88 C, A8 C, 90 C, 68 C, 00 C,
( R ) F0 C, 88 C, 88 C, F0 C, A0 C, 90 C, 88 C, 00 C,
( S ) 70 C, 88 C, 80 C, 70 C, 08 C, 88 C, 70 C, 00 C,
( T ) F8 C, 20 C, 20 C, 20 C, 20 C, 20 C, 20 C, 00 C,
( U ) 88 C, 88 C, 88 C, 88 C, 88 C, 88 C, 70 C, 00 C,
( V ) 88 C, 88 C, 88 C, 88 C, 88 C, 50 C, 20 C, 00 C,
( W ) 88 C, 88 C, 88 C, A8 C, A8 C, D8 C, 88 C, 00 C,
( X ) 88 C, 88 C, 50 C, 20 C, 50 C, 88 C, 88 C, 00 C,
( Y ) 88 C, 88 C, 50 C, 20 C, 20 C, 20 C, 20 C, 00 C,
( Z ) F8 C, 08 C, 10 C, 20 C, 40 C, 80 C, F8 C, 00 C,
( [ ) 78 C, 40 C, 40 C, 40 C, 40 C, 40 C, 78 C, 00 C,
( \ ) 00 C, 80 C, 40 C, 20 C, 10 C, 08 C, 00 C, 00 C,
( ] ) F0 C, 10 C, 10 C, 10 C, 10 C, 10 C, F0 C, 00 C,
( ^ ) 00 C, 00 C, 20 C, 50 C, 88 C, 00 C, 00 C, 00 C,
( _ ) 00 C, 00 C, 00 C, 00 C, 00 C, 00 C, 00 C, F8 C,

```

DECIMAL

```

CREATE BITS ( --- a1 )
128 C, 64 C, 32 C, 16 C, 8 C, 4 C, 2 C, 1 C,

: BIT ( N1 --- F1 )
BITS + C@ AND 0= 1+ ;

: LC>UC ( c -- )
DUP 96 128 WITHIN 32 AND - ;

VARIABLE OUT-CHAR ASCII # OUT-CHAR !
: OUTC! ( c --- )
\ modify the character for displaying the banner
\ modifie le caractère d'affichage de la bannière
OUT-CHAR ! ;
: BANNER ( a n -- )
BOUNDS 8 0
DO CR 2DUP
?DO I C@ 127 AND LC>UC 32 -
8* CHAR-MATRIX + J + C@
7 0
DO DUP I BIT
IF OUT-CHAR @
ELSE BL
THEN EMIT
LOOP DROP
LOOP
LOOP 2DROP ;

: DEMO ( --- ) \ print demonstration message
\ affiche un message de démonstration
219 OUTC! \ use the ASCII character number 219
\ utilise le caractère de code ASCII 219

DARK CR
ENGLISH ?\ " WELCOME" BANNER
ENGLISH ?\ " TO TF83" BANNER
FRENCH ?\ " BIENVENUE" BANNER
FRENCH ?\ " SUR TF83" BANNER
GERMAN ?\ " WILKOMMEN" BANNER
GERMAN ?\ " UBER TF83" BANNER
2000 MS
DARK CR
ENGLISH ?\ " BANNER" BANNER
ENGLISH ?\ " PROGRAM" BANNER

```

```

ENGLISH ?\ " FROM F83X" BANNER ;
FRENCH ?\ " PROGRAMME" BANNER
FRENCH ?\ " BANNIERE" BANNER
FRENCH ?\ " DE F83X" BANNER ;
GERMAN ?\ " BANNER" BANNER
GERMAN ?\ " PROGRAM" BANNER
GERMAN ?\ " VON F83X" BANNER ;
DEMO

```

MODE D'EMPLOI DU RESEAU BTX

par Marc PETREMANN

Ce sujet avait déjà fait l'objet d'un précédent article. Dans le présent propos, des précisions sont apportées concernant certaines manipulations.

Pour arriver à la page d'accueil de BTX, tout d'abord se connecter avec son Minitel à la passerelle permettant d'arriver sur le réseau allemand. Composer le 3622 ou 36224949 selon la zone d'appel; tarif 1,83 F/mn.

Un message s'affiche alors, indiquant que la connexion est en cours.

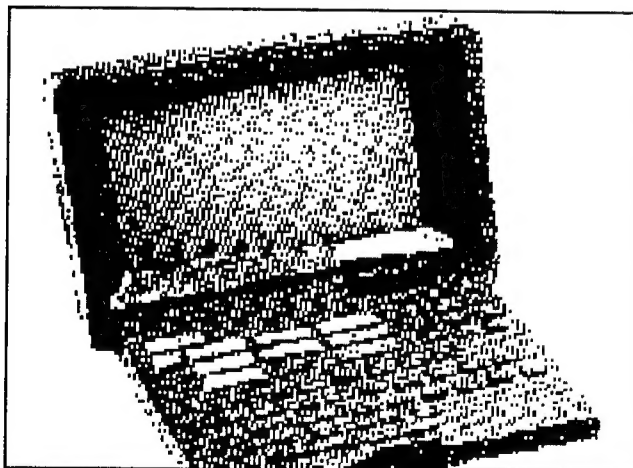
Lorsque BTX a accepté l'appel, tapez "#" pour afficher le sommaire général de BTX en allemand, ou "*1302#" pour afficher le sommaire en français.

Le choix 11 donne la liste des fournisseurs par ordre alphabétique, le choix 12 par mots-clés ou le choix 13 par thème.

Pour le moment, seules les pages gratuites sont accessibles.

On peut aussi appeler directement la page d'accueil du service souhaité. Exemple: pour appeler l'annuaire électronique des abonnés au téléphone en RFA, tapez le N de la page d'accueil du service entouré de * et # soit *1188# ou le choix 83 de la page *1302#.

On peut également arriver directement sur une page à l'intérieur d'un service.



Pour faciliter l'interrogation de BTX à partir d'un minitel, la passerelle BTX prend en compte l'utilisation de certaines touches de fonction de notre minitel:

- CONNEXION/FIN: met fin à la connexion
- SUITE: page suivante
- ENVOI: envoie un formulaire rempli au serveur (par contre, le choix dans un menu BTX ne nécessite pas l'appui sur la touche ENVOI comme dans Télétel; il suffit de taper sur le n°).

- REPETITION: réaffiche l'écran...
- Les autres touches de fonction du Minitel sont inactives.

Les touches de fonction dans BTX sont:

- *N de page#: appel d'une page
- *nom du serveur#: appel d'un service
- *#: retour page précédente
- #: page suivante
- ** : correction d'une donnée
- *00#: répétition d'une page
- *05#: affichage du contenu du champ de données...
- *09#: appel de la version courante d'une page
- *03#: retour au sommaire précédemment affiché
- *55#: retour à la page précédemment affichée du dernier fournisseur ou programme
- *1#: guide
- *0#: pour revenir au sommaire général de BTX
- *9#: connexion/fin

Quelques services; pour le moment, seules les pages gratuites sont accessibles depuis la France.

A titre d'exemple, voici quelques services du sommaire général de BTX. Pour accéder à leur page d'accueil, tapez:

- *1188#: l'annuaire électronique des abonnés au téléphone en RFA
- *20000#: Deutsche Bundespost (poste)
- *25800#: Deutsche Bundesbahn (chemins de fer)
- *30000#: Quelle (télé-achat)
- *41411#: Croix Rouge (action sociale)
- *50000#: Lufthansa (compagnie aérienne)
- *60000#: Deutsche Bank (banque)...

Contrairement au réseau TELETEL, les services ne sont pas répartis sur des serveurs à taxation par paliers d'utilisation (3613, 3614, 3615, etc...) mais pratiquent la taxation directe. Pour exemple, certains serveurs vous donnent accès au téléchargement de données. Ce téléchargement devient effectif après affichage du prix et envoi d'impulsions de taxation par le serveur. Or, la passerelle TELETEL/BTX filtrant ces impulsions, il ne vous est pas possible à ce jour de disposer de ces services.

De même, la passerelle TELETEL/BTX ne donne accès aux opérateurs appelant depuis un terminal relié à BTX aux seuls services accessibles en 3613 et 3614.

Une extension de l'accès aux services TELETEL étrangers est prévue sur les réseaux TELETEL italiens et belges.

STRUCTURE OUT-ENDOUT

par Yves SURREL

Adaptable: F83 Laxen et Perry CP/M et MS-DOS,
VolksForth ATARI.
Diffusion: 3615 SAM*JEDI et module 7 TURBO-Forth

LISTING: INOUT.FTH

```

\ *****
\
\          STRUCTURE OUT ... ENDOUT
\          STRUCTURE IN ... ENDIN
\ *****
\ Y. SURREL juin 1989
\
\ ----- OUT ... ENDOUT -----
\
\ Suivi de PERFORM, analogue à ON ... GOSUB du BASIC.
\ Utilisation OUT (S n -- ) ] MOTO MOT1 MOT2 ... MOTP

```



```

\ DEFAULT ENDOUT PERFORM
\ le (n+1)ième mot suivant OUT est exécuté.
\ Si n est supérieur à p,
\ DEFAULT est exécuté. n négatif est équivalent à n=0.
\ Peut dans certains cas remplacer avec une écriture
\ beaucoup plus simple une suite de OF ... END OF dans
\ une structure CASE ... ENDCASE.
\ Analogue à CASE: mais économise un identificateur (une
\ sorte de CASE: en ligne).
\ La structure (S n -- add ) OUT ... ENDOUT seule
\ délivre l'adresse de la (n+1)ième cellule de deux
\ octets suivant OUT. La compilation est arrêtée après
\ OUT pour permettre de remplir la table de données
\ entre OUT et ENDOUT avec autre chose que des CFAs en
\ utilisant la virgule:
\      2 OUT 5 , 8 , -1 , 3 , ENDOUT
\ empile -1 (les données sont numérotées à partir de 0).

```

```

: OUT@ (S n add2 add1 -- max[min[add2,add1+2n],add1] )
ROT 2* OVER + (S add2 add1 add )
MAX MIN ;

```

```

: OUT (S -- add )
\ add est l'adresse du début de la table d'exécution
COMPILE BRANCH
\ sauter inconditionnellement par dessus cette table
?>MARK [COMPILE] [ ; IMMEDIATE

```

```

: ENDOUT (S add -- )
DUP -ROT ?>RESOLVE
HERE 2- [COMPILE] LITERAL
2+ [COMPILE] LITERAL
COMPILE OUT@ ] ; IMMEDIATE

```

```

\ ----- IN ... ENDIN -----

```

```

\ Recherche une valeur empilée dans une table de n
\ valeurs numérotées de 0 à n. Retourne n en cas d'échec
\ de la recherche.

```

```

\ Une sorte de ASSOCIATIVE: en ligne.
\ Remarque importante: IN arrête le mode compilation
\ Exemples:

```

```

\ : EX (S n -- index ) IN 3 , 5 , -12 , ENDIN ;
\ 3 EX empilera 0
\ 5 EX " 1
\ 6 EX " 3

```

```

: IN@ (S n add cnt ) \ cf ASSOCIATIVE:
>R 0 -ROT R> 0 (S 0 n add cnt 0 )
DO (S i n add )

```

```

2DUP @ =

```

```

IF

```

```

LEAVE

```

```

THEN

```

```

2+ ROT 1+ -ROT

```

```

LOOP

```

```

2DROP ;

```

```

: IN (S -- add )
\ add est l'adresse du début de la table de valeurs
[COMPILE] OUT ; IMMEDIATE

```

```

: ENDIN (S add -- )

```

```
DUP -ROT ?>RESOLVE
2+ DUP [COMPILE] LITERAL
HERE 4 - SWAP - 2/ [COMPILE] LITERAL
COMPILE IN@ ] ; IMMEDIATE
```

```
\ -----
\ Exemple
\ -----
```

```
\ Réécriture de .PFA du décompilateur avec IN ... ENDIN
\ et OUT ... ENDOUT
\ Plus élégant que ASSOCIATIVE: et CASE:, à mon avis.
```

also HIDDEN also

\ Décompile une définition :

```
: .PFA (S CFA -- )
  DUP 'CFA ! >BODY .POS
  BEGIN
    #OUT @ 50 >
    IF .POS THEN
      DUP @
      IN ]
        ( 0 ) (LIT)      ( 1 ) ?BRANCH
        ( 2 ) BRANCH    ( 3 ) (LOOP)
        ( 4 ) (+LOOP)   ( 5 ) (DO)
        ( 6 ) COMPILE   ( 7 ) (")
        ( 8 ) (ABORT")  ( 9 ) (;CODE)
        ( 10 ) UNNEST   ( 11 ) (")
        ( 12 ) (?DO)    ( 13 ) (;USES)
      ENDIN
      OUT ]
        ( 0 ) .INLINE   ( 1 ) .BRANCH
        ( 2 ) .BRANCH   ( 3 ) .BRANCH
        ( 4 ) .BRANCH   ( 6 ) .BRANCH
        ( 6 ) .QUOTE    ( 7 ) .STRING
        ( 8 ) .STRING   ( 9 ) .(;CODE)
        ( 10 ) .UNNEST  ( 11 ) .STRING
        ( 12 ) .BRANCH  ( 13 ) .FINISH
        ( 14 ) .WORD
      ENDOUT PERFORM
      DUP 0= STOP? OR
    UNTIL DROP ;
```

' HELLO .PFA

VISUALISATION D'IMAGES DIGITALISEES

Par Yves SURREL

Adaptable: F83 Laxen et Perry CP/M et MS-DOS,
VolksForth ATARI.
Diffusion: 3615 SAM*JEDI et module 7 TURBO-Forth.

LISTING: DIGIT.FTH

```
\ Ce programme permet l'affichage de fichiers d'images
\ digitalisées au format .GRS délivrés par les routines
\ DGI1LIB (carte DGI1), mais il doit être adaptable
\ facilement à d'autres formats.
\ Ces fichiers sont constitués de 255*256 octets contigus
\ sans en-tête. Chaque octet contient le niveau de gris
\ d'un pixel sur 6 bits, le septième bit (40h) étant à 1.
\ Les pixels se suivent de gauche à droite et de haut en bas
\ (le premier octet du fichier correspond donc au coin
\ supérieur gauche de l'image).
\ L'affichage EGA ne fonctionnera bien entendu correctement
\ que si l'on dispose de la carte correspondante.
\ BUG connu: l'usage de CHARGE entraîne à la sortie de FORTH
```

```
\ l'apparition d'un message d'erreur: "Erreur
\ d'allocation mémoire. Impossible de charger
\ COMMAND.COM. Arrêt du système" avec RESET obligatoire.
\ Notre secrétaire pourra sans doute remédier à ceci...
```

```
ECHO OFF
FORTH DEFINITIONS DECIMAL
VARIABLE CARTE \ 1 = EGA 2 = CGA
VARIABLE 1er_OCTET
VARIABLE #OCTETS
\ Nombre d'octets à sauter entre 2 lignes
VARIABLE #LIGNES
VARIABLE OPERATION_LOGIQUE
CREATE TABLE_SEUILS 64 ALLOT
\ Correspondance niveaux de gris => couleur
VARIABLE POIGNEE
40 STRING FILES
```

```
HEX
VARIABLE SEG.IMAGE
: CHARGE
  EXT$ PAD PLACE
  " .GRS" EXT$ $!
  OPEN POIGNEE !
  SEG.IMAGE @ 0 FF00 POIGNEE @ (GET) ?DOS-ERR DROP
  CLOSE HERE COUNT FILES $! PAD COUNT EXT$ $! ;

: SAUVE (S <nom fichier [.GRS]> -- )
  ?OPEN DRV >R
  EXT$ PAD PLACE " .GRS" EXT$ $!
  \ Extension .GRS par défaut
  FILENAME PAD COUNT EXT$ $!
  DUP 1+ C@ ASCII : =
  \ Spécification de drive?
  IF
    DUP C@ ASCII A - SELECT
  THEN
    FF00. FREE D> ABORT" Espace insuffisant sur disque!"
    0 (CREATE) ?DOS-ERR \ (S hndl )
    >R
    SEG.IMAGE @ 0 FF00 R@ (PUT) \ Ecriture 8400h octets
    ?DOS-ERR DROP
    R> (CLOSE) R> SELECT ;
```

HEX

DEFER (AFFICHE)

```
CODE (AFFICHE.EGA) (S -- )
  STI
  IP PUSH
  RP PUSH
  3CE # DX MOV
  F01 # AX MOV
  AX DX OUT
  SI SI XOR
  1er_OCTET #) DI MOV \ Octet écran dans DI
  80 # DL MOV \ 1er pixel 1000 0000
  A000 # BP MOV \ RAM video...
  BP ES MOV \ ... dans ES
  #LIGNES #) CX MOV
  SEG.IMAGE #) AX MOV \ Segment IMAGE
  AX DS MOV HERE
  ( DO ) \ Pour chaque ligne FAIRE...
  CX PUSH
  \ Sauvegarder compteur de boucle externe
  100
  DO \ Pour chaque pixel FAIRE...
  CX PUSH
  \ Sauvegarder compteur de boucle interne
  AL BYTE LODS
  \ Niveau de gris OR 40h dans AL
  3F # AX AND
  AX BX MOV \ Niveau de gris dans BX
  CS: TABLE_SEUILS [BX] CL MOV
  \ Couleur dans CL
  \ Affiche du pixel DL de l'octet DI
```

```

DX PUSH          \ Sauvegarder position pixel
8 # AL MOV
DL AH MOV        \ Position pixel dans AH
3CE # DX MOV
AX DX OUT
AL AL XOR
CL AH MOV        \ Couleur dans AH
AX DX OUT
CS: OPERATION_LOGIQUE #) AX MOV
AL AH MOV
AH SHL
AH SHL
AH SHL
3 # AL MOV
AX DX OUT
ES: 0 [DI] AL MOV
ES: AL 0 [DI] MOV
DX POP

```

\ Fin affichage pixel

```

DL SHR          \ Pixel suivant
0 # DL CMP      \ Est-ce le dernier?
0=
IF              \ Si oui
    DI INC      \ incrémenter l'octet écran
    80 # DL MOV
    \ et remettre le compteur pixel sur 1000 0000

```

```

THEN
    CX POP      \ Dépiler compteur de boucle interne

```

LOOP \ Passer au pixel suivant

CS: #OCTETS

#) DI ADD \ Après 256 pixels, sauter n octets écran

80 # DL MOV \ Pixel 1000 0000

CX POP \ Dépiler compteur de boucle externe

LOOP \ Passer à la ligne suivante

3CE # DX MOV

1 # AX MOV

AX DX OUT

\ Rétablissement de l'affichage des caractères

FF0B # AX MOV

AX DX OUT

3 # AX MOV

AX DX OUT

\ Sortie

CS AX MOV

AX DS MOV

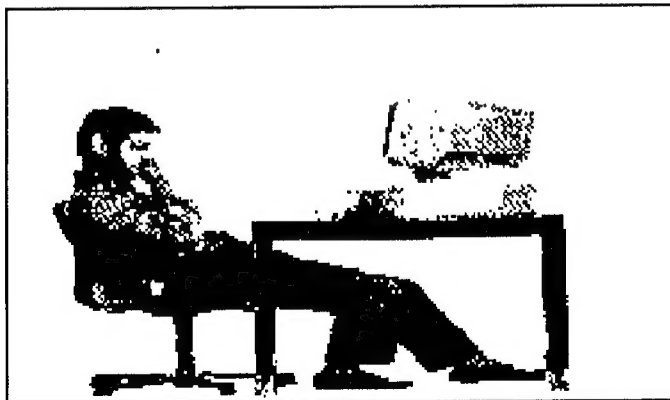
RP POP

IP POP

CLI

NEXT

END-CODE



CODE (AFFICHE.CGA)

STI

IP PUSH

8 # DI MOV

\ Octet écran dans DI

0 # SI MOV

\ 1er octet image

B800 # AX MOV

\ Segment vidéo

AX ES MOV

\ ... dans ES

SEG.IMAGE #) AX MOV

AX DS MOV

C8 DO \ 100 lignes paires: 0, 2, 4,...

CX PUSH \ puis 100 impaires.

20 DO \ 32 mots

CX PUSH \ soit 64 octets ou 64x4=256 pixels

8 DO \ Pour chaque mot de 8 pixels

AL LODS \ Charger le niveau de gris

3F # AX AND \ AND 3F

AX BX MOV \ Niveau de gris dans BX

CS: TABLE_SEUILS [BX] AL MOV \ Couleur dans AL

3 # AL AND \ Modulo 4

DX SHL \ décaler à gauche le pixel précédent

AL DX OR \ (couleur sur 2 bits...)

AL DX OR \ Mettre pixel dans DX

LOOP \ pixel suivant

DX AX MOV

AL AH XCHG \ Octet fort à gauche

AX STOS \ Afficher les 8 pixels

CX POP

LOOP

10 # DI ADD \ Saut de ligne écran

100 # SI ADD \ Saut de ligne image

C800 # SI CMP

\ Si les 100 lignes paires sont affichées

0= IF

100 # SI MOV

\ Se positionner sur les lignes impaires

1FB8 # DI MOV \ et la 2ème ligne écran

THEN

CX POP

LOOP

CS AX MOV

AX DS MOV

IP POP

CLI

NEXT

END-CODE

9000 SEG.IMAGE !

: EGA1

['] (AFFICHE.EGA) IS (AFFICHE)

1 CARTE !

18 1er_OCTET !

30 #OCTETS !

FF #LIGNES ! ;

: EGA2

['] (AFFICHE.EGA) IS (AFFICHE)

2 CARTE !

4 1er_OCTET !

8 #OCTETS !

C8 #LIGNES ! ;

: CGA

['] (AFFICHE.CGA) IS (AFFICHE)

8 1er_OCTET !

3 CARTE ! ;

: /NOP OPERATION_LOGIQUE OFF ;

: /AND 1 OPERATION_LOGIQUE ! ;

: /OR 2 OPERATION_LOGIQUE ! ;

: /XOR 3 OPERATION_LOGIQUE ! ;

DECIMAL

: AFFICHE

CARTE @

CASE

1 OF GRMODE @ 16 <> IF 16 MODE THEN END OF


```

2 OF GRMODE @ 13 <> IF 13 MODE THEN END OF
3 OF GRMODE @ 5 <> IF 5 MODE THEN END OF
TRUE ABORT" Carte non initialisée"
ENDCASE (AFFICHE) ;

: ORIGINE (S x y -- )      \ x et y en pixels
CARTE @ 1 =
IF
  95 MIN 80 *
  SWAP 383 MIN 8 / + 1er_OCTET !
THEN ;

: ASYST 375 1 ORIGINE ;

: DEFAULT \ Valeur des seuils par défaut
64 0
DO
  1 4 / TABLE_SEUILS I + C!
LOOP ; DEFAULT

: ?SEUILS
GRMODE @ DUP 2 <> SWAP 3 <> AND IF 3 MODE THEN
CR
5 SPACES 201 EMIT 64 205 REPLICATE 187 EMIT CR 5 SPACES
186 EMIT 25 SPACES ." NIVEAUX DE GRIS" 24 SPACES 186 EMIT
CR 5 SPACES
199 EMIT 64 196 REPLICATE 182 EMIT CR 5 SPACES
186 EMIT
." 0      1      2      3"
."      4      5      6  "
186 EMIT CR 5 SPACES 186 EMIT
6 0 DO ." 0123456789" LOOP ." 0123"
186 EMIT CR 5 SPACES 204 EMIT 64 205 REPLICATE 185 EMIT
CR 5 SPACES 186 EMIT
BASE @ >R HEX
64 0
DO
  TABLE_SEUILS I + C@ (.) TYPE
LOOP
186 EMIT CR 5 SPACES
200 EMIT 64 205 REPLICATE 188 EMIT CR
R> BASE ! ;

: SEUIL (S niveau1 niveau2 couleur -- )
DUP 16 <
IF
  -ROT 2DUP < NOT
  IF (S couleur niveau1 niveau2 )
    SWAP
  THEN
  DUP 64 <=
  IF
    OVER - (S couleur niveau1 plage )
    >R TABLE_SEUILS + R>
    ROT FILL
  ELSE TRUE ABORT" Niveau haut supérieur à 64!"
  THEN
  ELSE TRUE ABORT" Couleur supérieure à 16"
  THEN ;

\ Histogrammes

CREATE HIST.DATA 128 ALLOT

HEX

CODE (HISTOGRAM)
IP PUSH
SI SI XOR
1 # DX MOV
HIST.DATA 80 - # BX MOV
SEG.IMAGE #) AX MOV
AX DS MOV
AH AH XOR
FF00
DO
  AL LODS
  AL SHL

```

```

AX DI MOV
CS: DX 0 [BX+DI] ADD
LOOP
CS AX MOV
AX DS MOV
IP POP
NEXT
END-CODE

DECIMAL

: HISTOGRAM HIST.DATA 128 ERASE (HISTOGRAM) ;

```

TURBO RESIDENT

par Yves SURREL

Adaptable: F83 Laxen et Perry CP/M et MS-DOS,
VolksForth ATARI.
Diffusion: 3615 SAM*JEDI et module 7 TURBO-Forth

LISTING: HALT1.FTH

echo off
WARNING OFF

\ Modifié par Y.SURREL juin 1989.

ONLY FORTH ALSO HIDDEN ALSO DEFINITIONS HEX

VARIABLE _SP	VARIABLE _RP	\ Sauvegardes
VARIABLE _IP		\ des
VARIABLE _SS		\ registres

\ Structure de la pile lors de l'appel depuis ASYST
\ (cf p. G1-2-54):

\		
\ -	Off 2	-	Adresses hautes
\ -	Seg 2	-	\
\ -	Off 1	-	+E
\ -	Seg 1	-	+C
\ -	n	-	+A
\ -	Ret off	-	+8
\ -	Ret seg	-	+6
\ -	DS	-	+4
			\ paramètre empilé
			\ par ENTREE
			\ ci-dessous
\ -	ES	-	+2
\ -	Flags	-	SS:SP +0
			Adresses basses

CODE ENTREE

\ Point d'entrée pour l'appel par programme extérieur
(ASYST)

```

DS PUSH
ES PUSH
PUSHF
CS AX MOV
AX DS MOV
SS BX MOV
BX _SS #) XCHG
AX SS MOV
IP _IP #) XCHG \ Récupère la configuration
SP _SP #) XCHG \ existant au moment du dernier
RP _RP #) XCHG \ BYE ou HALT.
NEXT
END-CODE

```

```

CODE SORTIE
  \ Sortie dans le cas de l'appel à partir d'ASYST.
  \ Utilisé dans RETOUR ci-dessous.
  IP _IP #) XCHG
  SP _SP #) XCHG
  RP _RP #) XCHG
  _SS #) AX MOV
  AX SS MOV
  POPF
  ES POP
  DS POP
  0 FAR +RET
END-CODE

HERE 2- CONSTANT #.TO.POP
  \ sert à remplacer 0 FAR +RET par n FAR +RET

\ primitive de sortie avec programme laissé résident
CODE (HALT) ( taille-résidente-en-paragraphe -- )
  IP _IP #) MOV      \ Sauvegarde configuration
  SP _SP #) MOV      \
  RP _RP #) MOV      \
  DX POP
  3100 # AX MOV
  21 INT
  NEXT
END-CODE

\ modification de ALLOC pour n'importe quel PSP
LABEL RES-ALLOC ( taille -- dernier_seg )
  62 # AH MOV 21 INT BX ES MOV \ récupère PSP courant
  BX POP 4A # AH MOV 21 INT 1PUSH
RES-ALLOC ' ALLOC !

5F CONSTANT RES-INT
  \ n° de l'interruption de "retour en forth"

FORTH DEFINITIONS
: #PARAMS
  _SS @ _SP @ 0A + L@ ;

: PARAM ( S n -- seg off )
  \ Récupère les paramètres passés sur la pile _SS: _SP
  >R @ #PARAMS <= R@ 0 > AND
  IF
    R> NEGATE #PARAMS + 1+ >R
    _SS @ _SP @ 08 + R@ 4 * + L@ (S segment)
    _SS @ _SP @ 0A + R@ 4 * + L@ (S segment offset)
  THEN R> DROP ;

: RETOUR
  #PARAMS 2* 1+ 2* #.TO.POP ! SORTIE ;

CODE BYE
  IP _IP #) XCHG
  RP _RP #) XCHG
  SP _SP #) XCHG
  4C # AH MOV
  21 INT
END-CODE

\ sortie de Forth laissé résident avec mot de relance
: HALT ( <mot-de-relance> -- )
  BL WORD COUNT 80 PLACE
  \ un mot ou rien dans le PSP de Turbo
  0 RES-INT 4 * 2 + L@ D SEGMENT <>
  IF
    ['] ENTREE >BODY 0 RES-INT 4 * L!
    \ non : initialise vecteur sur Forth
    DSEGMENT 0 RES-INT 4 * 2+ L!
    HEIGHT DUP ALLOC DROP
    \ limite taille minimale du forth
    (HALT) DROP
    \ et sortie résidente
  ELSE
    BYE
  THEN ;

```

```

CR
.( commande HALT [<mot>] active pour sortie résidente )

\ création d'un micro-fichier .COM (19 octets)
\ de relance du Forth

:: filename 10 (search0)
  if only forth also decimal eof then ; RESUME.COM
\ cette commande interprétée stoppe ici la
\ compilation de ce fichier
\ si RESUME.COM existe déjà dans le répertoire courant.

CR
.( création d'un fichier RESUME.COM pour )
( revenir au Forth )
warning off
LABEL RESUME
  \ code du micro-programme de relance du forth
  0 # AX MOV AX DS MOV RES-INT 4 * 2+ #) AX MOV
  AX AX OR 0<> IF
    RES-INT INT \ exécute une interruption 50H
    THEN 4C00 # AX MOV 21 INT
    \ fin de fichier si pas de INT50

RESUME HERE SAVE RESUME.COM
FORGET RESUME

WARNING ON

ONLY FORTH ALSO DEFINITIONS DECIMAL
eof -----
Ce programme est une modification de HALT.FTH permettant
l'appel de FORTH résident par un autre langage
(personnellement, j'utilise ASYST) avec passage de
paramètres. La structure de la pile lors de l'appel est
décrite au début du programme. D'autres langages
utilisent la même structure de pile lors d'appel de
procédures extérieures (BASIC...).

Une autre modification concerne la sauvegarde du
contexte FORTH (piles). Cela autorise ceci:
  BYE MOT (sous DOS, RESUME lancera FORTH et MOT
sera exécuté)
ou:
  RETOUR MOT (le prochain appel par ASYST (p. ex.)
enchainera sur MOT)

J'utilise un FORTH résident en mode "esclave":

: SLAVE
  BEGIN
    1 PARAM L@ \ selon la valeur du 1er paramètre...
    CASE
      0 OF QUIT ENDOF \ quitter le mode esclave
      1 OF ACTION1 ENDOF \ ou exécuter
      2 OF ACTION2 ENDOF
        \ différents traitements....
      ....
    ENDCASE
    RETOUR \ ...vers ASYST. Attente d'une commande.
  AGAIN ;

ASYST

```

RETROUVEZ CES PROGRAMMES EN TELECHARGEMENT ASCII
 SUR NOTRE SERVEUR EN COMPOSANT LE 3615, CODE
 SAM*JEDI (accès sans restriction). SI VOUS APPELEZ
 DE L'ETRANGER, COMPOSEZ LE (33) 1-36.43.15.15
 SAM*JEDI (1200/75 bauds E71); 32 VOIES D'ACCES.

TOURS DE HANOI

par Peter Midnight

Adaptable: F83 Laxen et Perry CP/M et MS-DOS,
VolksForth ATARI.
Diffusion: 3615 SAM*JEDI et module 7 TURBO-Forth

LISTING: HANOI.FTH

\ The Famous Towers of Hanoi by Peter Midnight.
\ From FORTH Dimensions Volume 2 Number 2.
\ Converted to Laxen and Perry F83 by Jack Brown.
\ The original author left out the stack comments and
\ I don't have time to put them in today. Send me a
\ copy with them in!

```
12 CONSTANT NMAX VARIABLE (N) NMAX (N) !
: N ( -- n ) (N) @ ; \ fetch value of n.
219 CONSTANT COLOR
  VARIABLE HFO 3 HFO !
  VARIABLE RING N ALLOT

: DELAY ( n -- )
  0 DO 17 0 DO 127 127 * DROP LOOP LOOP ;

: POS ( n n' )
  N 2* 1+ * N + ;

: HALFD ( char size -- )
  0 DO DUP EMIT LOOP DROP ;

: <DISP> ( row char size -- )
  2DUP HALFD ROT 3 < IF BL ELSE ( ASCII H) 219
  THEN EMIT HALFD ;

: DISPLAY
  SWAP >R -ROT OVER - R@ AT
  R> -ROT <DISP> ;

: PRESENCE ( n flag )
  RING + C@ = ;

: LINE
  4 SWAP N 0 DO DUP I PRESENCE 1+
  ROT + SWAP LOOP DROP ;

: RAISE
  DUP POS SWAP LINE 2 SWAP
  DO 2DUP I BL DISPLAY 2DUP I 1- COLOR DISPLAY
  -1 +LOOP 2DROP ;

: LOWER
  DUP POS SWAP LINE 1+ 2
  DO 2DUP I 1- BL DISPLAY 2DUP I COLOR DISPLAY
  LOOP 2DROP ;

: MOVELEFT
  POS SWAP POS 1- DO DUP I 1+ 1 BL DISPLAY
  DUP I 1 COLOR DISPLAY -1 +LOOP DROP ;

: MOVERIGHT
  POS 1+ SWAP POS 1+ DO DUP I 1- 1 BL DISPLAY
  DUP I 1 COLOR DISPLAY LOOP DROP ;

: TRAVERS
  2DUP > IF MOVELEFT ELSE MOVERIGHT THEN ;

: MOVE1
  KEY? IF 0 16 AT ABORT THEN
  -ROT 2DUP RAISE >R 2DUP R> ROT TRAVERS
  2DUP RING + 1- C! SWAP LOWER ;

: MULTIMOV RECURSIVE
  3 PICK 1 = IF DROP MOVE1 ELSE
  >R >R SWAP 1- SWAP R> R> 4DUP SWAP MULTIMOV
  4DUP DROP ROT 1+ -ROT MOVE1
  -ROT SWAP MULTIMOV THEN ;

: MAKETOWER
  POS 4 N + 3 DO DUP I AT 219 EMIT LOOP DROP ;

: MAKEBASE
  0 N 4 + AT N 6 * 3 + 0 DO 219 EMIT LOOP ;

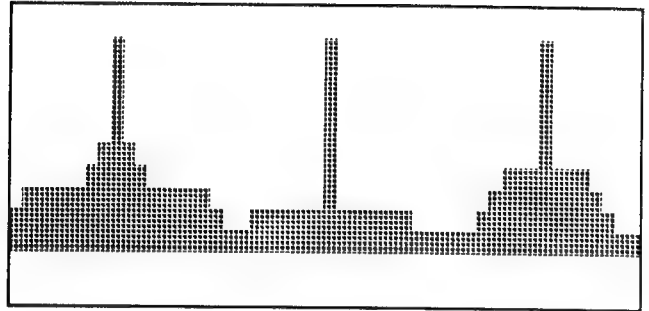
: MAKERING
  2DUP RING + 1- C! SWAP LOWER ;

: SETUP
```

DARK N 1+ 0 DO 1 RING I + C! LOOP 3 0 DO I
MAKETOWER LOOP MAKEBASE 1 N DO 0 I MAKERING -1
+LOOP ;

\ Values of n larger than 7 take a fair amount of time.
: TOWERS (n --)
 1 MAX NMAX MIN (N) !
 SETUP N 2 0 1 BEGIN
 OVER POS N 4 + AT
 ROT 4DUP MULTIMOV
 HFO @ 1- DUP HFO ! UNTIL
 2DROP 2DROP
 0 0 AT
 ;

: HANOI 7 TOWERS ;



JEU DE CAPTURE

par Bruce T. NICHOLAS

Adaptable: F83 Laxen et Perry CP/M et MS-DOS,
VolksForth ATARI.
Diffusion: 3615 SAM*JEDI et module 7 TURBO-Forth

LISTING: CAPTURE.FTH

\ FORTH DIMENSIONS VOL.X No.6 1988
\ (Turbo-Forth-83 Version by M.ZUPAN June 1989)

\ TIME OF DATE WORDS
\ -----

VARIABLE HOURS : @HRS HOURS @ ;
VARIABLE MINUTES : @MIN MINUTES @ ;
VARIABLE SECONDS : @SEC SECONDS @ ;

CODE GETIME \ Get the time of day
 AX POP AL AH MOV 33 INT CX PUSH
 AH AH SUB 2PUSH END-CODE
: TIME@ \ Get the time
 44 GETIME DROP 256 /MOD ROT 256 /MOD ;
: TIME! \ Store the time
 TIME@ HOURS ! MINUTES ! SECONDS ! DROP ;
: TIME \ Show elapsed time during game
 TIME@ 60 * + 60 * + @HRS 60 * @MIN + 60 * @SEC + -
 60 /MOD . ." Minutes and " . ." Seconds " DROP ;

\ CODE WORDS
\ -----

CODE POKE (byte seg off ---) \ Put a byte LC!
 BX POP ES POP AX POP ES: AL 0 [BX] MOV
 NEXT END-CODE
CODE PEEK (seg off --- byte) \ Get a byte LCA

```

AX AX SUB  BX POP  ES POP  ES: 0 [BX] AL MOV
1PUSH  END-CODE
CODE EQUIP  ( --- equip )  \ List of hardware
17 INT  1PUSH  END-CODE

```

```

HEX
: CGA?  ( --- VideoBuffer Attrib)
  \ B&W or Color video monitor
  EQUIP 30 AND 30 = IF B000 70
    ELSE B800 71 THEN ;

```

```

DECIMAL
CGA? CONSTANT ATTRIB
  CONSTANT SEGMENT

```

```

\ RANDOM GENERATOR
\ -----

```

```

VARIABLE SEED      TIME@ + + * SEED !

: RAND      ( -- random no. )
  SEED @ 5421 * 1+ DUP SEED ! ;
: RANDOM  ( n -- random no. ) \ Random number generator
  RAND FLIP SWAP MOD ;

```

```

\ EXTENDED KEYBOARD KEYS
\ -----

```

```

: EKEY
  KEY DUP 0= IF DROP KEY THEN ;
: WKEY
  EKEY 0= IF DROP THEN ;

```

```

\ MATRIX WORDS
\ -----

```

```

: MATRIX      ( #rows #columns -- )
  \ create a two dimensional array
  CREATE
    2DUP , ,
    * 2* ALLOT ;

: ELEMENT      ( row# col# `matrix -- `element )
  DUP >R @
  ROT 2* * SWAP 2* +
  R> 4 + + ;

```

```

24 80 MATRIX SCREEN \ Build data image of the screen

```

```

: FIND-LENGTH ( addr -- length )
  \ find the length of the data matrix
  SCREEN DUP 2+ @ SWAP @ * ;
: FILL-ARRAY ( array-index -- )
  \ Fill the matrix with data
  0 DO 1 2* 0 I SCREEN ELEMENT ! LOOP ;

```

```

\ GAME VARIABLES
\ -----

```

```

VARIABLE LEVEL
VARIABLE SPEED
VARIABLE #BEASTS
VARIABLE END
VARIABLE XKEY
VARIABLE CNT
VARIABLE X
VARIABLE Y
VARIABLE WIN
VARIABLE LOSE
VARIABLE DIFFCNT
VARIABLE ANOTHER
1 END !
1 LOSE !

```

```

CREATE H 8 ALLOT
CREATE V 8 ALLOT

```

```

\ TITLE AND INSTRUCTIONS
\ -----

```

```

: TITLE      \ Display the title page on the screen
  31 1 AT ." CAPTURE"
  20 4 AT ." Trap the beasts before they get you!"

```



```

23 6 AT ." Box them in completely to win."
20 8 AT ." Use the numeric keypad to move
HERO." ;

```

```

: KEYS      \ Show the cursor control keys
  31 11 AT ." 7          9 " 38 11 AT 30 EMIT
  31 12 AT ."          "
  31 13 AT ."          "
  31 14 AT ."  ----  ---- " 32 14 AT 17 EMIT
  31 15 AT ."          " 44 14 AT 16 EMIT
  31 16 AT ."          "
  31 17 AT ." 1          3 " 38 17 AT 31 EMIT
  23 20 AT ." Tap any other key to stop HERO."
  23 22 AT ." Press ESC key to quit anytime."

```

```

;
: CONTINUE \ Pauses the display
  25 24 AT ." Press any KEY to continue." WKEY ;
: INVERSE \ Reverse video for title
  3 0 DO
    30 0 DO ATTRIB SEGMENT
      63 I + 160 J * + POKE 2 +LOOP
    LOOP ;
: INVERSE1 \ Reverse video for keypad area
  9 0 DO
    34 0 DO ATTRIB SEGMENT 1661 I +
      160 J * + POKE 2 +LOOP
    LOOP ;
: ASK      \ Read the key and limit the input
  KEY 48 - 1 MAX MIN DUP 48 + EMIT ;
: #BEASTS? \ Store the number of beasts
  20 20 AT ." Number of beasts ( 1, 2, or 3 ) ? "
  3 ASK #BEASTS ! ;
: DIFFICULTY? \ Store the difficulty level
  20 22 AT
    ." Level of difficulty ( 1 to 3 Easiest ) ? "
  3 ASK LEVEL ! ;
: SPEED?   \ Store the speed of the game
  20 24 AT ." Set speed ( 1 to 9 slowest ) ? "
  9 ASK SPEED ! ;

```

```

\ BORDERS AND COLORS
\ -----

```

```

: THERE      ( h v -- seg addr )
  \ Location to put the screen data
  SWAP SEGMENT -ROT SCREEN ELEMENT @ ;
: T&B      \ Draw the top and bottom
  79 1 DO 205 I 0 THERE POKE
    205 I 23 THERE POKE LOOP ;
: SIDES     \ Draw sides of the display box
  23 1 DO 186 0 I THERE POKE
    186 79 I THERE POKE LOOP ;
: CORNERS   \ Draw the four corners
  201 0 0 THERE POKE
  200 0 23 THERE POKE
  187 79 0 THERE POKE
  188 79 23 THERE POKE ;
: BORDERS   \ Draw a complete box around
  T&B SIDES CORNERS ;
: COLOR     \ Set screen attribute bytes
  3840 1 DO 23 SEGMENT I POKE 2 +LOOP ;

```

```

\ OBSTACLES BEAST HERO and misc
\ -----

```

```

: HORZ      78 RANDOM 1+ ;
: VERT      22 RANDOM 1+ ;
: HB        H CNT @ 2* + ;
: VB        V CNT @ 2* + ;

: OBSTACLES \ Place obstacles on screen
  400 0 DO 177 HORZ VERT THERE POKE LOOP ;
: VACANT?   \ Check if the location is empty
  BEGIN HORZ DUP HB !
  VERT DUP VB !
  THERE PEEK BL =
  UNTIL ;
: BEAST     \ Place the number of beasts wanted
  -1 CNT !
  BEGIN CNT 1+! VACANT?

```



```

200 0 DO BL HB @ VB @ THERE POKE
42 HB @ VB @ THERE POKE
LOOP
CNT @ #BEASTS @ 1- =
UNTIL ;

: V+1 V 8 + @ 1+ ;
: V-1 V 8 + @ 1- ;
: H+1 H 8 + @ 1+ ;
: H-1 H 8 + @ 1- ;
: V0 V 8 + @ ;
: H0 H 8 + @ ;
: H1 H 8 + ;
: V1 V 8 + ;

: OPOKE \ Put the data to the screen
H0 V0 THERE POKE ;

: HERO \ Look for a place to put our HERO
4 CNT ! VACANT?
200 0 DO BL OPOKE 2 OPOKE LOOP ;

: VACANT \ Is location blank?
DUP BL = ;

: UP^ \ Move one space up
V0 0 DO
H0 V-1 I - THERE PEEK VACANT IF
OPOKE 177 H0 V-1 I - THERE POKE
2 H0 V-1 THERE POKE
V-1 V1 ! LEAVE ELSE
42 = IF LEAVE THEN THEN LOOP ;
: DOWN \ Move one space down
23 V0 - 0 DO
H0 V+1 I + THERE PEEK VACANT IF
OPOKE 177 H0 V+1 I + THERE POKE
2 H0 V+1 THERE POKE
V+1 V1 ! LEAVE ELSE
42 = IF LEAVE THEN THEN LOOP ;
: LEFT \ Move one space left
H0 0 DO
H-1 I - V0 THERE PEEK VACANT IF
OPOKE 177 H-1 I - V0 THERE POKE
2 H-1 V0 THERE POKE
H-1 H1 ! LEAVE ELSE
42 = IF LEAVE THEN THEN LOOP ;
: RIGHT \ Move one space right
79 H0 - 0 DO
H+1 I + V0 THERE PEEK VACANT IF
OPOKE 177 H+1 I + V0 THERE POKE
2 H+1 V0 THERE POKE
H+1 H1 ! LEAVE ELSE
42 = IF LEAVE THEN THEN LOOP ;
: UP_LEFT \ Move one space up and left
V0 0 DO
H-1 I - V-1 I - THERE PEEK VACANT IF
OPOKE 177 H-1 I - V-1 I - THERE POKE
2 H-1 V-1 THERE POKE
H-1 H1 ! V-1 V1 ! LEAVE ELSE DUP
42 = IF DROP LEAVE THEN
186 = IF LEAVE THEN THEN LOOP ;
: UP_RIGHT \ Move one space up and right
V0 0 DO
H+1 I + V-1 I - THERE PEEK VACANT IF
OPOKE 177 H+1 I + V-1 I - THERE POKE
2 H+1 V-1 THERE POKE
H+1 H1 ! V-1 V1 ! LEAVE ELSE DUP
42 = IF DROP LEAVE THEN
186 = IF LEAVE THEN THEN LOOP ;
: DOWN_LEFT \ Move one space down and left
23 V0 - 0 DO
H-1 I - V+1 I + THERE PEEK VACANT IF
OPOKE 177 H-1 I - V+1 I + THERE POKE
2 H-1 V+1 THERE POKE
H-1 H1 ! V+1 V1 ! LEAVE ELSE DUP
42 = IF DROP LEAVE THEN
186 = IF LEAVE THEN THEN LOOP ;
: DOWN_RIGHT \ Move one space down and right

```

```

23 V0 - 0 DO
H+1 I + V+1 I + THERE PEEK VACANT IF
OPOKE 177 H+1 I + V+1 I + THERE POKE
2 H+1 V+1 THERE POKE
H+1 H1 ! V+1 V1 ! LEAVE ELSE DUP
42 = IF DROP LEAVE THEN
186 = IF LEAVE THEN THEN LOOP ;
: FUNCTION> \ Check for the cursor key and move
DUP 72 = IF DROP UP^ EXIT THEN
DUP 80 = IF DROP DOWN EXIT THEN
DUP 77 = IF DROP RIGHT EXIT THEN
DUP 75 = IF DROP LEFT EXIT THEN
DUP 71 = IF DROP UP_LEFT EXIT THEN
DUP 73 = IF DROP UP_RIGHT EXIT THEN
DUP 79 = IF DROP DOWN_LEFT EXIT THEN
DUP 81 = IF DROP DOWN_RIGHT EXIT THEN
DUP 27 = IF DROP END OFF EXIT THEN
DROP 2 OPOKE ;

: MOVE_HERO \ Move the HERO. He can push obstacles
KEY? IF EKEY DUP XKEY ! ELSE XKEY @ THEN FUNCTION> ;

: Y=0 4 RANDOM 1- 2/ Y ! ;
: X=0 4 RANDOM 1- 2/ X ! ;
: Y<>0
O> IF -1 Y ! ELSE 1 Y ! THEN 100 RANDOM DUP
60 < IF DROP ELSE
75 > IF 0 Y ! ELSE Y @ NEGATE Y ! THEN THEN ;
: X<>0
O> IF -1 X ! ELSE 1 X ! THEN 100 RANDOM DUP
70 < IF DROP ELSE
80 > IF 0 X ! ELSE X @ NEGATE X ! THEN THEN ;
: MOVEB
HB @ H0 - DUP 0= IF DROP X=0 ELSE X<>0 THEN
VB @ V0 - DUP 0= IF DROP Y=0 ELSE Y<>0 THEN
HB @ X @ + VB @ Y @ + THERE PEEK
BL = IF BL HB @ VB @ THERE POKE
42 HB @ X @ + VB @ Y @ + THERE POKE
HB @ X @ + HB !
VB @ Y @ + VB !
THEN ;
: MOVE_BEAST \ Move the beasts
-1 CNT !
BEGIN CNT 1+! MOVEB CNT @ #BEASTS @ 1- = UNTIL ;

: DELAY \ Speed of game loop
25 SPEED @ 9 * * 0 DO LOOP ;
: SET_LEVEL \ Set game difficulty level
DIFFCNT DUP @ 1- DUP ROT ! ;

\ LOSE? and WIN?
\ -----
: LOSE? \ Check for loosing the game
-1 CNT !
BEGIN CNT 1+!
3 0 DO 3 0 DO
HB @ I 1- + VB @ J 1- + THERE PEEK
2 = IF 0 END !
4 24 AT ." YOU LOSE " 0 LOSE ! LEAVE
ELSE 1 LOSE !
THEN
LOOP LOOP
CNT @ #BEASTS @ 1- =
UNTIL ;

: WIN? \ Check for winning the game
-1 CNT ! 0 WIN !
BEGIN CNT 1+!
3 0 DO 3 0 DO
HB @ I 1- + VB @ J 1- + THERE PEEK
32 = IF LEAVE
ELSE WIN 1+!
THEN
LOOP LOOP
CNT @ #BEASTS @ 1- =
UNTIL
WIN @ #BEASTS @ 9 * =
IF 4 24 AT ." YOU WIN " 0 END ! THEN ;

```

```

: Y/N      \ Check for a YES or NO answer
BEGIN KEY UPC DUP 78 = IF 1 ANOTHER ! DROP EXIT THEN
          89 = IF -1 ANOTHER ! THEN
          ANOTHER @ 0 <>
UNTIL ;

```

```

\ CAPTURE
\ -----
: TELL \ Display the instructions
DARK TITLE INVERSE
KEYS INVERSE1
CONTINUE 20 24 AT 32 SPACES 20 22 AT 40 SPACES
#BEASTS? DIFFICULTY? SPEED?
FIND-LENGTH FILL-ARRAY ;

```

```

: PLAY \ Draw the playing board & play
1 END ! 0 XKEY ! 0 ANOTHER ! DARK
COLOR BORDERS OBSTACLES BEAST HERO
CONTINUE 20 24 AT 32 SPACES TIME!
LEVEL @ DIFFCNT !
BEGIN   DELAY MOVE_HERO
        SET_LEVEL
        0= IF MOVE_BEAST LEVEL @ DIFFCNT ! THEN
        LOSE? LOSE @
        0= IF ELSE WIN? THEN
        50 24 AT TIME
        END @ 0=
UNTIL 20 24 AT ;

```

```

: GAME      \ Play the game of CAPTURE
BEGIN TELL PLAY 50000 0 DO LOOP
      20 24 AT ." Play again (Y/N) "
      Y/N ANOTHER @ 1 = UNTIL DARK ;

```

EOF
 \ To build a .COM file of the program:

```

: CAPTURE GAME BYE ;
' CAPTURE IS BOOT
SAVE-SYSTEME CAPTURE.COM
EOF

```

This game is based on the program originally written in Applesoft BASIC by Rob Smythe for the Apple II and published in NIBBLE Magazine Vol.2/No.4/1981. I wrote this version as a learning experience.

Bruce T. Nicholas

Ce jeu a été initialement écrit en BASIC Applesoft par Rob Smythe pour Apple II et publié dans le magazine NIBBLE Vol.2/N°4/1981. J'ai écrit cette version à titre d'expérience d'apprentissage.

DESASSEMBLEUR 8086 POUR TF83

par Charles CURLEY
 Adaptation TURBO-FORTH-83 par Michel Zupan 06/89

Adaptable: F83 Laxen et Perry MS-DOS.
 Diffusion: 3615 SAM*JEDI et module 7 TURBO-Forth

```

LISTING:      DISASS86.FTH
\ adr DIS : désassemble à partir d'une adresse
              intra-segment
\ seg adr LDIS : désassemble à partir d'une adresse
              extra-segment
\ n IDIS : désassemble l'interruption n
\ NDIS : reprend un désassemblage au dernier point d'arrêt
\ SEE <mot> : revectorisé pour désassembler un mot en code
\ ECHO @ ECHO OFF

```

ONLY FORTH ALSO ROOT DEFINITIONS
 VOCABULARY DISASSEMBLER
 DISASSEMBLER ALSO DEFINITIONS DECIMAL
 WARNING OFF

```

: EXEC: 2* R> + PERFORM ;

```

```

CODE 2/S      \ n ct --- n' | shift n right ct times
CX POP
AX POP
AX CL SHR
1PUSH
END-CODE

```

```

CODE 2*S      \ n ct --- n' | shift n left ct times
CX POP
AX POP
AX CL SHL
1PUSH
END-CODE

```

```

CODE SEXT
\ n --- n' | sign extend lower half of n to upper
AX POP
CBW
1PUSH
END-CODE

```

```

2VARIABLE RELOC
DSEGMENT 0 RELOC 2! \ keeps relocation factor
: T@ RELOC 2@ ROT + L@ ;
\ in first word, seg in 2nd. You
\ dump/dis any segment w/ any
: TC@ RELOC 2@ ROT + LC@ ;
\ relocation you want by setting
\ RELOC correctly.

```

```

VARIABLE CP
VARIABLE SAVEBASE BASE @ SAVEBASE !

: OOPS CR SAVEBASE @ BASE ! TRUE ABORT" OOPS!" ;

: NEXTB CP @ TC@ CP 1+! ;
: NEXTW CP @ T@ 2 CP +! ;

: .MOI
\ --- | have the current word print out its name.
LAST @ [COMPILE] LITERAL COMPILE .ID ; IMMEDIATE

```

```

VARIABLE OPS \ operand count
VARIABLE IM \ 2nd operand extension flag/ct

```

```

: ?DISP
\ op ext --- op ext | does MOD operand have a disp?
DUP 6 2/S DUP 3 = OVER 0= OR
0= IF IM ! ELSE
0= IF DUP 7 AND 6 = IF 2 IM ! THEN THEN THEN ;

```

```

: .SELF
\ -- | create a word which prints its name
CREATE LAST @ , DOES> @ .ID ;
\ the ultimate in self-doc!

```

.SELF AL	.SELF AX	.SELF [BX+SI]	.SELF ES
.SELF CL	.SELF CX	.SELF [BX+DI]	.SELF CS
.SELF DL	.SELF DX	.SELF [BP+SI]	.SELF SS
.SELF BL	.SELF BX	.SELF [BP+DI]	.SELF DS
.SELF AH	.SELF SP	.SELF [SI]	.SELF #
.SELF CH	.SELF BP	.SELF [DI]	.SELF #)
.SELF DH	.SELF SI	.SELF [BP]	.SELF S#)
.SELF BH	.SELF DI	.SELF [BX]	
.SELF RP	.SELF [RP]	\ RETURN STACK POINTER	
.SELF IP	.SELF [IP]	\ INTERPRETER POINTER	
.SELF W	.SELF [W]	\ WORKING REGISTER	

```

VARIABLE SYMBOLIC SYMBOLIC ON

```

```

6 CONSTANT SYMBOLCT

```

```

CREATE SYMBOLS ASSEMBLER
>NEXT , >NEXT 1- , >NEXT 2- , >NEXT 3- ,
' BRANCH >BODY
' (LOOP) 5 + ,
DISASSEMBLER
.SELF NEXT .SE LF 1PUSH .SELF 2PUSH
.SELF 3PUSH
.SELF BRAN1 .SE LF PLOOP
: ?SYMBOL
\ a -- a n | if n = -1 then no symbol, else index
TRUE SYMBOLIC @ IF \ if in code segment.
SYMBOLCT 0 DO OVER I 2* SYMBOLS + @ =
IF DROP I LEAVE THEN LOOP THEN ;

VARIABLE ENDDIS
: .SYMBOL \ a --- | print symbol name else value
?SYMBOL DUP 0< IF DROP U. EXIT THEN
ENDDIS ON SWAP U. EXEC:
NEXT 1PUSH 2PUSH 3PUSH BRAN1 PLOOP ;

: SYMBOL CREATE ' >NAME , ' >NAME ,
DOES> SYMBOLIC @ IF 2+ THEN @ .ID ;

SYMBOL BX BX W SYMBOL [BX] [BX] [W]
SYMBOL SI SI IP SYMBOL [SI] [SI] [IP]
SYMBOL BP BP RP SYMBOL [BP] [BP] [RP]

: .16REG \ r# --- | register printed out
7 AND EXEC: AX CX DX BX SP BP SI DI ;

: .8REG \ r# --- | register printed out
7 AND EXEC: AL CL DL BL AH CH DH BH ;

: .SEG \ s# --- | register printed out
3 2/S 3 AND EXEC: ES CS SS DS ;

: ODISP \ --- | do if displacement is 0
." 0 " ;

: BDISP \ --- | do if displacement is byte
CP @ IM @ + TC@ SEXT U. OPS 1+! IM OFF ;

: WDISP \ --- | do if displacement is word
CP @ IM @ + T@ U. 2 OPS +! IM OFF ;

: (.R/M) \ op ext --- | print a register
SWAP 1 AND IF .16REG ELSE .8REG THEN IM OFF ;

: .R/M \ op ext --- op ext | print r/m as register
2DUP (.R/M) ;

: .REG \ op ext --- op ext | print reg as register
2DUP 3 2/S (.R/M) ;

: .DISP \ op ext --- op ext | print displacement
DUP 6 2/S 3 AND EXEC: ODISP BDISP WDISP .R/M ;

: BIMM \ --- | do if immed. value is byte
CP @ IM @ + TC@ . 1 OPS +! IM OFF ;

HEX
: .MREG \ op ext --- op ext | register(s) printed out +
disp
DUP C7 AND 6 = IF WDISP #) ELSE
DUP C0 AND C0 - IF .DISP
DUP 7 AND EXEC: [BX+SI] [BX+DI] [BP+SI] [BP+DI]
[SI] [DI] [BP] [BX]
ELSE .R/M IM OFF THEN THEN ;

DECIMAL .SELF BYTE .SELF WORD
: .SIZE \ op --- | decodes for size
1 AND EXEC: BYTE WORD ;

CREATE SEGTB ASCII E C, ASCII C C, ASCII S C, ASCII D C,
: SEG: \ op --- | print segment overrides
3 2/S 3 AND SEGTB + C@ EMIT ." S:" ;

: POP \ op --- | print pops
DUP 15 = IF OOPS THEN .SEG .MOI ;

: PUSH \ op --- | print pushes
.SEG .MOI ;

: P/P \ op --- | pushes or pops
DUP 1 AND EXEC: PUSH POP ;

.SELF DAA .SELF DAS .SELF AAA .SELF AAS

: ADJUSTS \ op --- | the adjusts
3 2/S 3 AND EXEC: DAA DAS AAA AAS ;

: P/SEG \ op --- | push or seg overrides
DUP 5 2/S 1 AND EXEC: P/P SEG: ;

: P/ADJ \ op --- | pop or adjusts
DUP 5 2/S 1 AND EXEC: P/P ADJUSTS ;

: OGP \ op --- op | opcode decoded & printed
DUP 4 AND IF DUP 1 AND
IF WDISP ELSE BIMM THEN #
1 AND IF AX ELSE AL THEN ELSE
NEXTB OVER 2 AND
IF .MREG .REG ELSE ?DISP .REG .MREG
THEN 2DROP THEN ;

.SELF ADD .SELF ADC (.SELF AND, ) .SELF
XOR
(.SELF OR, ) .SELF SBB .SELF SUB .SELF
CMP
: AND, ." AND " ;
: OR, ." OR " ;

: OGROUP \ op --- | select 0 group to print
DUP OGP 3 2/S 7 AND EXEC:
ADD OR, ADC SBB AND, SUB XOR CMP ;

: LOWS \ op --- | 0-3f opcodes printed out
DUP 7 AND EXEC:
OGROUP OGROUP OGROUP OGROUP
OGROUP OGROUP P/SEG P/ADJ ;

: .REGGP \ op --- | register group defining word
CREATE LAST @ , DOES> @ SWAP .16REG .ID ;

.REGGP INC .REGGP DEC .REGGP PUSH .REGGP
POP

: POP \ op --- | handle illegal opcode for cs pop
DUP 56 AND 8 = IF ." illegal," DROP ELSE POP THEN ;

: REGS \ op --- | 40-5f opcodes printed out
DUP 3 2/S 3 AND EXEC: INC DEC PUSH POP ;
.SELF O .SELF NO
.SELF B/NAE .SELF NB/AE
.SELF E/Z .SELF NE/NZ
.SELF BE/NA .SELF NBE/A
.SELF S .SELF NS
.SELF P/PE .SELF NP/PO
.SELF L/NGE .SELF NL/GE
.SELF LE/NG .SELF NLE/JG

: .BRANCH \ op --- | branch printed out w/ dest.
NEXTB SEXT CP @ + .SYMBOL ASCII J EMIT 15 AND EXEC:
O NO B/NAE NB/AE E/Z NE/NZ BE/NA NBE/A
S NS P/PE NP/PO L/NGE NL/GE LE/NG NLE/JG ;

: MEDS \ op --- | 40-7f opcodes printed out
DUP 4 2/S 3 AND EXEC:
REGS REGS OOPS .BRANCH ;

: 80/81 \ op --- | secondary at 80 or 81
NEXTB ?DISP OVER 1 AND
IF WDISP ELSE BIMM THEN # .MREG
SWAP .SIZE 3 2/S 7 AND EXEC:
ADD OR, ADC SBB AND, SUB XOR CMP ;

```

```

: 83S      \ op --- | secondary at 83
NEXTB ?DISP BMM # .MREG
SWAP .SIZE 3 2/S 7 AND EXEC:
ADD OR, ADC SBB AND, SUB XOR CMP ;

: 1GP      \ op --- | r/m reg opcodes
CREATE LAST @ ,
DOES> @ >R NEXTB ?DISP .REG .MREG 2DROP
R> .ID ;

1GP TEST      1GP XCHG      .SELF LEA      .SELF MOV

: MOVRR/REG NEXTB ?DISP .REG .MREG 2DROP MOV ; \ 88-89

: MOVD      NEXTB      .MREG .REG 2DROP MOV ; \ 8A-8B

HEX
: MOVSM      \ op --- | display instructions 8C-8E
NEXTB OVER 8D = IF .MREG .REG LEA ELSE
OVER 8F = IF .MREG [ ' POP >NAME ] LITERAL .ID ELSE
SWAP 1 OR SWAP \ 16 bit moves only, folks!
OVER 2 AND IF .MREG DUP .SEG ELSE
( ?DISP ) DUP .SEG .MREG THEN MOV THEN THEN 2DROP ;

: 8MOVS      \ op --- | display instructions 80-8F
DUP 2/ 7 AND EXEC: 80/81 83S TEST XCHG
MOVRR/REG MOVD MOVSM MOVSM ;

DECIMAL
.SELF XCHG      .SELF CBW      .SELF CWD      .SELF CALL
.SELF WAIT      .SELF PUSHF     .SELF POPF     .SELF SAHF
.SELF LAHF

: INTER      \ --- | decode interseg jmp or call
NEXTW .SYMBOL ." : " NEXTW U. ;

: CALLINTER \ --- | decode interseg call
INTER CALL ;

: 9HIS      \ op --- | 98-9F decodes
7 AND EXEC:
CBW CWD CALLINTER WAIT PUSHF POPF SAHF LAHF ;

: XCHGA      \ op --- | 98-9F decodes
DUP 7 AND IF AX .16REG XCHG ELSE DROP ." NOP " THEN ;

: 90S      \ op --- | 90-9F decodes
DUP 3 2/S 1 AND EXEC: XCHGA 9HIS ;

.SELF MOVSM      .SELF CMPS

: MOVSM      \ op --- | A4-A5 decodes
.SIZE MOVSM ;

: CMPS      \ op --- | A6-A7 decodes
.SIZE CMPS ;

: .AL/AX      \ op --- | decodes for size
1 AND EXEC: AL AX ;

: MOVSM/ACC \ op --- | A0-A3 decodes
DUP 2 AND IF .AL/AX WDISP #) ELSE
WDISP #) .AL/AX THEN MOV ;

.SELF TEST      .SELF STOS      .SELF LODS      .SELF SCAS

: .TEST      \ op --- | A8-A9 decodes
DUP 1 AND IF WDISP ELSE BMM THEN # .AL/AX TEST ;

: STOS      ( op --- ) .SIZE STOS ;
: LODS      ( op --- ) .SIZE LODS ;
: SCAS      ( op --- ) .SIZE SCAS ;

: AOS      \ op --- | A0-AF decodes
DUP 2/ 7 AND EXEC:
MOVSM/ACC MOVSM/ACC MOVSM      CMPS
.TEST      STOS      LODS      SCAS ;

: MOVSM/IMM \ op --- | B0-BF decodes
DUP 8 AND IF WDISP # .16REG ELSE BMM # .8REG
THEN MOV ;

: HMEDS      \ op --- | op codes 80 - C0 displayed
DUP 4 2/S 3 AND EXEC: 8MOVS 90S AOS MOVSM/IMM ;

.SELF LES      .SELF LDS      .SELF INTO      .SELF IRET

: LES/LDS \ op --- | les/lds instruction C4-C5
NEXTB .MREG 1 SWAP .REG 2DROP 1 AND EXEC: LES
LDS ;

: RET \ op --- | return instruction C2-C3, CA-CB
DUP 1 AND 0= IF WDISP ." SP+" THEN
8 AND IF ." FAR " THEN .MOI ;

: MOVRR/M \ op --- | return instruction C2-C3, CA-CB
NEXTB ?DISP OVER 1 AND IF WDISP ELSE BMM THEN
#
.MREG OVER .SIZE MOV 2DROP ;

: INT      \ op --- | int instruction CC-CD
1 AND IF NEXTB ELSE 3 THEN U. .MOI ;

: INTO/IRET \ op --- | int & ired instructions CE-CF
1 AND EXEC: INTO IRET ;

: COS      \ op --- | display instructions C0-CF
DUP 2/ 7 AND EXEC:
OOPS RET LES/LDS MOVRR/M OOPS RET INT INTO/IRET
;

: AAS
\ op --- | does anybody actually use these things?
CREATE LAST @ , DOES> @ .ID NEXTB 2DROP ;

AAS AAM      AAS AAD

.SELF ROL      .SELF ROR
.SELF RCL      .SELF RCR
.SELF SHL/SAL   .SELF SHR
.SELF SAR

: SHIFTS \ op --- | secondary instructions d0-d3
DUP 2 AND IF CL THEN
NEXTB .MREG NIP 3 2/S 7 AND EXEC:
ROL ROR RCL RCR SHL/SAL SHR OOPS SAR ;

: XLAT      DROP .MOI ;

: ESC
\ op ext --- op ext | esc instructions d8-DF
2DUP .MREG [ HEX ] 38 AND [ DECIMAL ] SWAP 7 AND OR
U. .MOI ;

DEFER ESCCODE ' ESC IS ESCCODE

: DOS      \ op --- | display instructions D0-DF
DUP 8 AND IF NEXTB ESCCODE 2DROP EXIT THEN
DUP 7 AND EXEC:
SHIFTS SHIFTS SHIFTS SHIFTS AAM AAD OOPS XLAT ;

.SELF LOOPE/Z ( .SELF LOOP, )
.SELF JCXZ      .SELF LOOPNE/NZ
: LOOP, ." LOOP " ;

: LOOPS \ op --- | display instructions E0-E3
NEXTB SEXT CP @ + .SYMBOL 3 AND EXEC:
LOOPNE/NZ LOOPE/Z LOOP, JCXZ ;

.SELF IN      .SELF OUT      .SELF JMP

: IN/OUT \ op --- | display instructions
E4-E6,EC-EF
DUP 8 AND IF
DUP 2 AND IF .AL/AX DX OUT ELSE

```



```

        DX .AL/AX IN THEN ELSE
        DUP 2 AND IF .AL/AX BIMM # OUT ELSE
        BIMM # .AL/AX IN THEN THEN ;

: CALLS \ op --- | display instructions E7-EB
  DUP 2 AND IF DUP 1 AND
  IF NEXTB SEXT CP @ + .SYMBOL \ short
    ELSE INTER THEN ELSE NEXTW CP @ + .SYMBOL THEN
  3 AND EXEC: CALL JMP JMP JMP ;

: EOS \ op --- | display instructions E0-EF
  DUP 2 2/S 3 AND EXEC: LOOPS IN/OUT CALLS IN/OUT ;

: FTEST \ op --- | display instructions F6,7:0
  ?DISP OVER 1 AND IF WDISP ELSE BIMM THEN #
  .MREG DROP .SIZE TEST ;

.SELF NOT .SELF NEG .SELF MUL .SELF IMUL
.SELF DIV .SELF IDIV .SELF REP .SELF REPNE
.SELF LOCK .SELF HLT .SELF CMC .SELF CLC
.SELF STC .SELF CLI .SELF STI .SELF CLD
.SELF STD .SELF INC .SELF DEC .SELF PUSH

: MUL/DIV \ op ext --- | secondary instructions F6,7:4-7
  .MREG NIP 3 2/S 3 AND EXEC: MUL IMUL DIV IDIV ;

: NOT/NEG \ op ext --- | secondary instructions F6,7:2,3
  .MREG SWAP .SIZE 3 2/S 1 AND EXEC: NOT NEG ;

: F6-F7S \ op --- | display instructions F6,7
  NEXTB DUP 3 2/S 7 AND EXEC:
  FTEST OOPS NOT/NEG NOT/NEG
  MUL/DIV MUL/DIV MUL/DIV MUL/DIV ;

: FES \ op --- | display instructions FE
  NEXTB .MREG BYTE NIP 3 2/S 1 AND EXEC: INC DEC ;

: FCALL/JMP \ op ext --- | display call instructions FF
  .MREG 3 2/S DUP 1 AND
  IF S#) ." FAR " ELSE #) THEN NIP
  2/ 1 AND EXEC: JMP CALL ;

: FPUSH \ op ext --- | display push instructions FF
  DUP 4 AND IF .MREG 2DROP PUSH EXIT THEN OOPS ;

: FINC \ op ext --- | display inc/dec instructions FF
  .MREG NIP 3 2/S 1 AND EXEC: INC DEC ;

: FFS \ op --- | display instructions FF
  NEXTB DUP 4 2/S 3 AND EXEC:
  FINC FCALL/JMP FCALL/JMP FPUSH ;

: FOS \ op --- | display instructions F0-FF
  DUP 15 AND DUP 7 AND 6 < IF NIP THEN EXEC:
  LOCK OOPS REPNE REP HLT CMC F6-F7S F6-F7S
  CLC STC CLI STI CLD STD FES FFS ;

: HIGHS \ op -- | op codes C0 - FF displayed
  DUP 4 2/S 3 AND EXEC: COS DOS EOS FOS ;

: (INST) \ op --- | highest level vector table
  255 AND DUP 6 2/S 3 AND
  EXEC: LOWS MEDS HMEDS HIGHS ;

: INST \ --- | display opcode at ip, advancing as needed
  SYMBOLIC @ 0= IF RELOC 2+ @ (U.) TYPE ASCII : EMIT THEN
  CP @ 4 U.R
  2 SPACES
  NEXTB (INST) OPS @ CP +! OPS OFF IM OFF ;

: (DUMP) \ addr ct --- | dump as pointed to by reloc
  [ FORTH ]
  SPACE BOUNDS DO 1 TC@ 0 <# # # BL HOLD #> TYPE LOOP ;

```

ALSO FORTH DEFINITIONS

```

: LDIS \ seg addr --- | disassemble
  2 ?ENOUGH CP ! DUP RELOC 2+ !

```

```

DSEGMENT = IF SYMBOLIC ON ELSE SYMBOLIC OFF THEN
BASE @ SAVEBASE ! HEX ENDDIS OFF
BEGIN CP @ >R
  CR INST R> CP @ OVER - 35 ?TAB (DUMP)
  STOP? ENDDIS @ OR UNTIL
  SAVEBASE @ BASE ! ;

```

```

: DIS \ addr --- | disassemble in Forth segment
  DSEGMENT SWAP LDIS ;

```

```

: NDIS
  \ --- | continue disassembly from previous address
  RELOC 2+ @ CP @ LDIS ;

```

```

: IDIS \ n --- | disassemble interrupt routine
  255 AND 4 * 0 TUCK OVER 2+ L@ -ROT L@ LDIS ;

```

HIDDEN DEFINITIONS

```

: (SEE/DIS) \ cfa ---
  DUP @ OVER >BODY =
  IF CR DUP >NAME .ID
    \ Nota: compiler POLYGLOT.FTH, module 5
    \ le cas échéant.
    EXIST? FRENCH ?\ ." est en code machine"
    EXIST? ENGLISH ?\ ." is low level code"
    EXIST? GERMAN ?\ ." In Maschinensprache"
  >BODY DIS
  ELSE ((SEE))
  THEN ;

```

```

' (SEE/DIS) IS (SEE)

```

ONLY FORTH ALSO DEFINITIONS DECIMAL

```

MARK EMPTY
WARNING ON
ECHO !
EOF

```

```

=====
DESASSEMBLEUR 8086 DE CHARLES CURLEY
=====

```

Charles CURLEY est informaticien FORTH chez MAXTOR après avoir implémenté des systèmes FORTH sur 6502, 6809, LSI-11, 80x8x, 680xx pour des sociétés comme Rockwell International, Jet Propulsion Lab, Hughes Aircraft, Strand Lighting et d'autres. Il est membre du Silicon Valley Forth Interest Group et c'est à ce titre qu'il a versé dans le domaine public ce désassembleur.

L'adaptation que j'en ai faite pour TURBO-FORTH-83 porte sur des points de détail: mnémoniques sans virgule (MOV pour MOV,), suppression du désassemblage pas-à-pas au profit d'un affichage avec pauses en utilisant STOP?. J'ai introduit aussi un arrêt automatique du désassemblage des mots en code et le mot NDIS qui permet de poursuivre le désassemblage au delà de la dernière adresse désassemblée. Je me suis permis de supprimer l'affichage en fin de ligne des caractères ASCII correspondant au 'dump' de la ligne.

Le programme original de C. Curley comporte quelques bogues au niveau de l'interprétation de certaines instructions. Certaines ont été signalées par Bill MUENCH et j'en ai découvert certaines à l'usage. J'ai apporté donc une dizaine de corrections (INC/DEC 8reg, XCHGA-NOP, REP/REPNE, POP, ESC,LDS, MUL/DIV etc) mais j'ignore si quelqu'un a déjà eu le courage de vérifier de façon systématique si toutes les instructions sont correctement interprétées dans tous les modes d'adressage valides...

Le programme ajoute 4 mots de syntaxe simple dans le vocabulaire FORTH:

```

DIS ( adr -- ) pour désassembler dans le segment
Forth

```

```

LDIS ( seg adr -- ) pour désassembler extra-segment

```

IDIS (NumInt --) pour désassembler une interruption
 NDIS (--) reprend l'un quelconque des trois précédents
 et modifie le comportement de SEE pour que soient
 désassemblés les mots codés. NDIS permet de poursuivre ce
 désassemblage au delà d'une instruction de saut sur
 l'interpréteur interne, laquelle n'est pas toujours la fin
 du code d'un mot.

Quelques remarques sur le programme:

Notez l'intéressant mot d'exécution indicée EXEC: : dans une
 définition, n EXEC: exécute le n+1 ième mot après EXEC: (le
 premier mot étant exécuté pour l'indice n=0). La suite 'i
 EXEC: mot0 mot1...motN ;' doit obligatoirement terminer la
 définition du mot en cours. EXEC: ne dispose pas d'un
 contrôle d'intervalle de l'indice d'exécution.

L'usage de mots qui s'affichent eux-mêmes .SELF et .MOI (en
 français dans le texte!) provoque des homonymies. Si vous
 utilisez le vocabulaire DISASSEMBLER, méfiez-vous des mots
 BL # XOR OUT CALL NOT WORD... J'ai évité les homonymies sur
 AND OR et LOOP.

Il peut être utile de récupérer le désassemblage d'une
 portion de code dans un fichier. L'édition de ce fichier
 facilitera le ré-assemblage par l'assembleur Forth.

Voici la procédure à utiliser:

INCLUDE PIP	\ utilitaire de redirection
ATTRIBUTS OFF	\ évite codes ANSI parasites
PRN TO ESSAI.DIS	\ redirige printer sur fichier
PRINTING ON	\ sortie écran et imprimante
xxx yyy LDIS	\ désassemblage
PRINTING OFF	\ fin d'impression
RESTORE	\ fin de redirection

Michel ZUPAN Juin 89

CREATION DE LISTES DE VARIABLES ET DE CONSTANTES

par Ch. MOHR

Adaptable: F83 Laxen et Perry MS-DOS ou CP/M,
 VolksFORTH ATARI.
 Diffusion: 3615 SAM*JEDI et module 7 TURBO-Forth

LISTING: PREFUNI.FTH

dark

```

*****
\ *      ***  TURBO - FORTH  ***
\ *  UTILITAIRES PERMETTANT LA DESIGNATION D'UNE LISTE DE *
\ *  VARIABLES OU DE CONSTANTES SOUS UN PREFIXE UNIQUE *
*****
\ ( echo off )
\ En programmant on peut être amené à désigner en en-tête
\ une liste importante de variables ou de constantes. Taper
\ X fois " VARIABLE " ou " CONSTANT " peut dans ce cas
\ devenir fastidieux. Comme je suis paresseux, j'ai imaginé
\ quelques mots permettant, à l'instar du PASCAL, la
\ désignation sous un seul préfixe d'un train de variables
\ ou de constantes, c'est à dire:
\
\ VAR:  désignation de variables du type VARIABLE
\ 2VAR: désignation de variables du type 2VARIABLE
\ CONST: désignation de constantes du type CONSTANT
\ 2CONST: désignation de constantes du type 2CONSTANT

```

\ STR: désignation de variables chaînes de caractères
 \ du type STRING du TURBO-FORTH.

\ REMARQUES IMPORTANTES :

\ - Les différents types doivent être désignés sur une
 \ ligne (sans CR), le point-virgule final (code 59) ne
 \ servant que de simple délimiteur.
 \ - L'indication de la valeur chiffrée, lorsqu'elle est
 \ requise, ne peut se faire qu'en " LITTERAL " (16 ou
 \ 32 bits selon le cas).

echo off

only forth also definitions

\ VARIABLES NUMERIQUES

```

: var:
source ascii ; scan 0= swap 1- c@ bl <> or
\ verification de la présence
abort" Absence du délimiteur ;"
\ du délimiteur ; ( ascii 59 ).
begin source drop >in @ + c@
\ examen dans le flot d'entrée
\ des codes ascii à interpréter.
case
  bl of 1 >in +! true endof
  \ si code 32 se contente d'incrémenter
  \ le compteur d'interprétation + flag vrai
  ascii ; of false endof
  \ si code 59 prépare fin de boucle
  \ en laissant un flag faux.
  ascii ( of 41 parse 2drop true endof
  \ pour pouvoir insérer des commentaires.
  variable true swap
  \ sinon création de la variable suivante
  \ dans le flot d'entrée.
endcase
while repeat
  \ recommence tant que flag vrai.
  1 >in +! ;
  \ pour ignorer le délimiteur final.

```

```

: 2var:
source ascii ; scan 0= swap 1- c@ bl <> or
abort" Absence du délimiteur ;"
begin source drop >in @ + c@
case
  bl of 1 >in +! true endof
  ascii ; of false endof
  ascii ( of 41 parse 2drop true endof
2variable true swap
endcase
while repeat
  1 >in +! ;

```

\ CONSTANTES NUMERIQUES

```

: const:
source ascii ; scan 0= swap 1- c@ bl <> or
abort" Absence du délimiteur ;"
begin source drop >in @ + c@
case
  bl of 1 >in +! true endof
  ascii ( of 41 parse 2drop true endof
  ascii ; of false endof
  bl word number drop
  constant true swap
  endcase
while repeat
  1 >in +! ;

: 2const:
source ascii ; scan 0= swap 1- c@ bl <> or
abort" Absence du délimiteur ;"
begin source drop >in @ + c@

```

```

    case
      bl of 1 >in +!      true endof
      ascii ; of          false endof
      ascii ( of 41 parse 2drop true endof
    bl word number
    2constant true swap
    endcase
  while repeat
    1 >in +! ;

\ VARIABLES CHAINES DE CARACTERES.

: str:
source ascii ; scan 0= swap 1- c@ bl <> or
abort" Absence du delimitateur ;"
begin source drop >in @ + c@
  case
    bl of 1 >in +!      true endof
    ascii ; of          false endof
    ascii ( of 41 parse 2drop true endof
    bl word number
    string true swap
  endcase
while repeat
  1 >in +! ;

forth definitions decimal

echo on

\ EXEMPLES D'UTILISATION.

\ VAR:   XX YY ZZ TT UU WW ;
\ 2VAR:  2X ( commentaire ) 3X ( commentaire ) 4X ;
\ CONST: 100 CA 200 CB 300 CD ;
\ 2CONST: 10200. DCA 8000. DCB 3.1416 PI ;
\ STR:   40 A$ ( commentaire ) 50 B$ 100 C$ 4 D$ (
commentaire ) ;

\ ... et que le FORTH soit avec vous ...

eof

```

CONTENU DU FORUM SAM*JEDI

SECRETAIRE Du 09.06.89 A 19h04
DISPONIBILITE DU MODULE 8: CONTIENT EN 90 Ko TOUT LE PACKAGE DE VIRGULE FLOTTANTE CORDIC (EGALEMENT DIFFUSE EN TELECHARGEMENT).

DISPONIBLE AVANT LE MODULE 7, ON BOUSCULE L'ORDRE... ET EN VERSION ANGLAISE POUR LE MOMENT.

PRIX: 37 FR. A COMMANDER A ASSOCIATION JEDI
17, RUE DE LA LANCETTE 75012 PARIS

(PAIEMENT EN TIMBRES DE PREFERENCE POUR TOUTE COMMANDE INFERIEURE A 70 FR, SVP)

FORTH7 Du 09.06.89 A 21h17
A new type of data: VALUE

Partant du constat que les variables sont plus souvent lues par @ qu'ecrites par !, certains proposent une nouvelle forme de variable appelee VALUE qui fonctionnerait a l'execution comme une constante (donc sans le fetch @).

En Turbo-Forth, nous avons appele ca une 'pseudo-constante' declaree par CONSTANT et modifiable par IS:

```

2 CONSTANT DEUX
3 IS DEUX

```

Voici une implementation de VALUE dont l'avantage est de ne pas necessiter la definition d'une foule de mots pour

l'affectation en notation prefixee

```

: VALUE      ( <nom> -- )
0 constant ; \ comme une VARIABLE

: (VAL)      ( -- adr )
R@ @ >body R> 2+ >R ; ( run-time)

: VAL        ( <nom> -- adr )
state @ if compile (VAL)
else ' >body
then ; immediate

```

VALUE TEST definit une 'valeur' TEST
2 VAL TEST ! met 2 dans TEST
VAL TEST OFF remet TEST a zero
VAL TEST 1+! incremente la valeur TEST. TEST seul empile directement le contenu de la value et VAL TEST son adresse.

ALORS POUR OU CONTRE LES VALUE's? Ces VALUE's dechainent les passions aux US: heresie non standard pour les uns, la fin des litanies de @ pour d'autres, hybridation constantes-variables contre nature ou genial paradoxe forthien?

Notez qu'une VALUE est plus rapide que VARIABLE s'il s'agit d'une valeur qu'on n'affecte qu'exceptionnellement. C'est bien le sens d'une (pseudo-)constante.

Qu'en pensez-vous? Si vous avez l'intention de remplacer toutes vos VARIABLE's par des VALUE's je vous refilerai (VAL) en code...

EPICURE Du 10.06.89 A 22h01
V A L U E
ENTIA NON SUNT MULTIPLICANDA PRAETER NECESSITATEM.

LE TYPE 'VALUE' S'IMPOSE-T-IL?

SI SON INTERET EST EVIDENT SURTOUT LORSQU'IL S'AGIT DE VALEURS AFFECTEES EXCEPTIONNELLEMENT, LES CONSTANTES EXISTENT POUR CELA, ET SONT MODIFIABLES PAR LE PROCEDE EXCEPTIONNEL:

```
' CONSTANT-NAME >BODY !
```

SI LE BUT DE CE NOUVEAU TYPE EST D'EVITER LES @ ALORS IL EST REMARQUABLE QUE L'ON RESSENTE QUAND MEME LE BESOIN D'AVOIR ACCES A L'ADRESSE DE LA DONNEE, ET DE SURCROIT PAR 'VAL' QUI FAIT PENSER A TOUT SAUF A SON ADRESSE...

BOUSCULER LES HABITUDES ET, CE QUI EST AUTREMENT PLUS GRAVE ET LOURD DE CONSEQUENCES, CHANGER LE STANDARD, POUR UN RESULTAT SI LEGER, NE PARAIT PAS ESSENTIEL.
AVETE ATQUE VALETE.

SECRETAIRE Du 12.06.89 A 14h08
JE VIENS DE METTRE EN TELECHARGEMENT UN PREMIER PROTO DE L'ASSEMBLEUR F32.

ATTENTION: TELECHARGEMENT EN BINAIRE CAR CONTIENT DES CARACTERES ACCENTUES. JE PLANCHE SUR SIMULATEUR F32.

FORTH7 Du 14.06.89 A 21h16
Assemble ce qui, poussiere, se disperse
Lao Tseu

Le type VALUE malgre les critiques auxquelles il se prete, forth a considerer l'usage d'une syntaxe prefixee.

Il ne faut pas grand chose pour changer l'assembleur actuel postfixe de facon a ce qu'il comprenne les deux syntaxes.

Vous pourrez bientot ecrire au choix:
POSTFIX ES: BP SI MOV
ou PREFIX MOV ES: BP, SI

Ceux qui ont repris des fichiers MASM comprendront l'interet...

FORTH7 Du 14.06.89 A 21h21
La plus grande vertu s'ignore
Et c'est la raison de sa vertu
encore Lao Tseu (*)

Avez-vous lu le Science et Vie Micro de juin 89? Surtout ne l'achetez pas! (y'a rien dedans) mais lisez seulement chez votre buraliste l'entrefilet sur le FORTH page 188. Navrant.

(*) Sans rire, le Tao Te King (6eme s. av JC) est le livre de chevet d'un nombre incroyable de forthiens! Un peu fadas peut-etre ces fanas? Je vous ai du moins epargne la V.O. moi...

SECRETAIRE Du 15.06.89 A 08h50
Il est effectivement interessant de pouvoir selectionner un assemblage en mode pre-fixe ou post-fixe. Mais il ne faut pas generaliser ce mecanisme en FORTH. La tentation est grande de programmer des fonctions POST-FIXEES ou IN-FIXEES. Pour exemple, l'addition pourrait tres bien etre reecrite de maniere a etre exploitee ainsi:
n1 + n2 en interpretation et en compilation.
Mais ce serait dommageable pour la compacite et les performances de FORTH.

En fait, on pourrait, a force de suggestions, recreer un pseudo C ou PROLOG ou tout ce que vous voulez.

Les notations particulieres (non FORTHiennes) sont a reserver aux extensions particulieres destinees a des applications specifiques.

Et puis, la notation FORTH est bien supportee par POSTSCRIPT, alors que ces memes defenseurs de POSTSCRIPT n'ont aucune connaissance en FORTH (ceux qui gribouillent dans SVM par exemple... et contre qui j'ai une dent; affaire personnelle...).

Les nouveaux FORTH 32 bits seront certainement plus simples, mais auront des fonctionnalites etendues:
- fonctions virgule flottante,
- fonctions de gestion de fichiers
- fonctions graphiques,
- interface de communication,
etc...

Pour le moment, s'il n'existe pas de vrai FORTH 32 bits sur PC, c'est parce qu'une telle implementation se ferait au detriment des performances. Les micro-processeurs INTEL, par leur technique de segmentation de la memoire, rendent ce probleme ardu. Sur un micro-processeur de la gamme MOTOROLA, une implementation de FORTH 32 bits est plus aisee. Mais entre MacINTOSH, ATARI, AMIGA, ou tout autre systeme equipe d'un 68xxx, il n'y a pas deux DOS semblables.

Un programmeur souhaitant implementer FORTH en 32 bits sur ce type de machines en est reduit a faire un langage minimal s'il ne dispose pas de toutes les adresses du DOS et du systeme. Sans renier la superiorite des capacites et performances des systemes equipes en 68xxx, ce qui fait la force des systemes IBM, est:
- ils sont globalement plus repandus,
- la compatibilite logicielle est ascendante PC->XT->AT,
- ils tournent pour la plupart avec MSDOS.

Donc, au risque de decevoir ceux qui attendent une eventuelle mouture amelioree de TURBO-Forth, nous leur confirmons que la version actuelle restera encore exploitable dans sa forme actuelle pendant un certain temps.

EPICURE Du 15.06.89 A 22h33
Le ma Citre des anciens temps
Etait subtil perspicace merveilleux et profond.
(Lao Tseu)

Forth est par essence postfixe, la seule exception etant les chaines (pour eviter la confusion, dit Chuck Moore [Forth Dimensions VIII/1] et encore invite-t-il des suggestions pour supprimer cette entorse...). Faire du prefixe, comme le secretaire le dit presque, ce sont voies de brigandage
mais non pas la Voie.
(Lao Tseu encore)
AVETE ATQUE VALETE

SECRETAIRE Du 16.06.89 A 16h34
PROGRAMME DE TELES AISIE PAR MINITEL:
Il avance tres bien. Maintenant, toutes les donnees sont transferees sur IBM dans un fichier dBASE.

Sur MINITEL, les champs de saisie sont geres comme peuvent l'etre ceux de dBASE. Pour ce faire, j'ai eu l'idee suivante:

```
5 STRING A$
5 STRING B$
5 STRING C$
3 CASE: VAL$ A$ B$ C$ ;
et un mot READ demande la saisie sequentielle des
donnees A$ B$ et C$ quand on compile READ:
```

```
: TEST READ VAL$ ;
```

Toute zone de saisie MINITEL passe en video inverse; si elle est deja en video inverse, on rajoute la surintensite. Miam, miam...

Maintenant, si vous ne voulez pas partager ces idees avec les votres, peu importe; moi je gere une saisie telematique et un fichier dBASE en moins de 40 Ko. Essayez avec CLIPPER de faire mieux... cauchemar garanti! Allez, bon week-end a tous.

FORTH7 Du 19.06.89 A 19h43
EARLY BINDING VERSUS LATE BINDING. Les mots d'execution differee du Forth permettent d'expliquer tres facilement la difference entre le chainage precoce et le chainage tardif, notions au coeur des langages objets et de l'IA.

Considerez le simple mot:

```
: [DEFERRED] ( <MOT> -- )
STATE @ IF ' >IS @ , THEN ; IMMEDIATE
```

qui s'emploie dans une definition pour compiler le mot DEFERred qui le suit:

```
: ... [DEFERRED] EMIT ... ;
```

BINDING BINDING BINDING (air connu). Vous avez compris le mot? Bravo!

[DEFERRED] EMIT va compiler non pas EMIT mais son vecteur tel que defini AU MOMENT DE LA COMPILATION. C'est ca le 'early binding' mon chou. Tandis que lorsque vous compilez EMIT tout seul, vous faites reference AU MOMENT DE L'EXECUTION et que voila du 'late binding' sans le savoir, Mr Jourdain...

Y'a que Forth (et moi) pour trouver des trucs si simples.

DEFER GASTON. Vous allez me dire que [DEFERRED] ne sert a rien: compiler (EMIT) d'emblee est plus simple. A premiere vue, oui.

J'ai defini ici [DEFERRED] dans un but pedagogique en prenant DEFER comme base connue. La dynamique du Forth-Orient-Objet utilise des mots vectorises qui compilent directement en early binding: c'est le concept de METHOD aussi novateur que l'a ete DEFER. Si ! (store) est une Methode, il compilera C! W! L! \$! array! etc selon l' OBJET reference.

' BYE IS GASTON

PYLA Du 22.06.89 A 09h04
POSSEDE UN SUPER SHAREWARE (selfmade!) POUR TELECHARGER DES
PROGS SUR SAM*JEDI. IL EST POUR VOUS:*** GRATUITEMENT ***
COMMANDEZ-LE SUR 3615 SER*PC, rubrique pyla soft
A BIENTOT ET BONS TELECHARGEMENTS

PS: IL PERMET AUSSI LA CAPTURE D'ECRANS MINITEL (sa premiere
vocation a l'origine)

FF32 Du 22.06.89 A 15h04
POST-OU-PREFIXE? LA QUESTION EST COMMENT UTILISER UN LANGAGE
PUISSANT COMME FORTH QUI SOIT PRESENTABLE COMME UN VRAI
LANGAGE ?

LA CONTROVERSE FAIT RAGE ENTRE LAO-TSE, GASTON ET EPICURE...
POUR MA PART, QUE J'AIMERAIS POUVOIR COMPRENDRE LES PETITS
MOTS DE FORTH7 SANS AVOIR BESOIN DE DESSINER LA PILE A
CHACQUE MOT. MAIS JE NE SUIS QU'UN ETRE HUMAIN ORDINAIRE...

ALORS JE VOUS FABRIQUE UNE MACHINE EXTRA POUR DESSINER VOS
MOTS AVEC UNE SOURIS; ON MONTRERA AUX CURIEUX LES !@ QUI
SONT DESSOUS, S'ILS SONT COURAGEUX. F32 QUI VOUS VEUT DU
BIEN.

SECRETAIRE Du 22.06.89 A 18h22
FORTH ET C: JE PARAITRAI PEUT ETRE IGNARE MAIS JE VIENS DE
DECOUVRIR QUE C TRANSMET SES DONNEES ENTRE ROUTINES PAR LA
PILE...

ALORS PEUT-ETRE SERIT-IL POSSIBLE D'EXECUTER DU CODE COMPILE
EN C DEPUIS FORTH. JE DIS CA COMME CA, HISTOIRE DE RIGOLER,
POUR REMPLIR LA PAGE.

SECRETAIRE Du 23.06.89 A 16h51
Mon KEY MINITEL revu et corrige:
HEX FFFF CONSTANT TIME-OUT
VARIABLE #ESSAIS
CODE (KEY) (--- c)
TIME-OUT # AX MOV #ESSAIS # BX MOV
AX 0 [BX] MOV 03FD # DX MOV
BEGIN DX AL IN 01 # AL TEST 0=
WHILE 0 [BX] AX MOV AX DEC 0=
IF 1PUSH
ELSE AX 0 [BX] MOV
THEN
REPEAT
03F8 # DX MOV DX AL IN AH AH XOR
1PUSH END-CODE

Et voila, (KEY) attend qu'un caractere parvienne de la ligne
MINITEL sur le port serie du PC. Si aucun caractere n'est
recu apres 0,5 sec, (KEY) empile le code 0, sinon, il empile
le code du caractere recu.

Ce programme peut egalement etre utilise dans un fichier
.BAT:

- remplacer les 1PUSH par RET
- taper apres la compilation de (KEY):
' (KEY) 2+ HERE SAVE RSKEY.COM

BYE

Dans un fichier .BAT, RSKEY renvoie le code du caractere
recu dans AX lequel est analyse par ERRORLEVEL.
Exemple:

```
mode com1:1200
:debut
rskey
if errorlevel = 0 goto debut
echo VOUS ETES CONNECTE > com1:
echo > com1:
echo 1..PROGRAMME TRUC > com1:
echo 2..PROGRAMME MACHIN > com1:
:redo
rskey
if errorlevel = 51 goto redo
```

```
if errorlevel = 50 goto machin
if errorlevel = 49 goto truc
if errorlevel = 48 goto redo
: machin
copy machin.txt com1:
goto debut ...etc...
```

Ce petit programme permet de gere un serveur vraiment
minimal.

Pour l'exploiter sur site a 4800 bauds, taper sur votre
minitel 1B:

```
FNCT-P-4
FNCT-T-A (mode 80 colonnes)
et sur le PC:
MODE COM1:4800
```

Ouf, c'etait vraiment dur!!!

FORTH7 Du 24.06.89 A 22h41
IL Y A MILLE FACONS D'OUBLIER ...
FORGET oublie radicalement un mot et tous ceux qui l'ont
suivi.

FORBID oublie un mot mais conserve son code pour tous
les mots compiles apres lui. Un exemple en est donne
dans le nouvel assembleur F32 pour "oublier" l'ancien.

Voici le petit dernier:

REMOVE deplace un mot dans un autre vocabulaire, HIDDEN
par exemple.

Utile si le mot ne doit pas etre perdu.

```
VARIABLE REMOTE \ pointe le vocabulaire de destination
' HIDDEN IS REMOTE
: REMOVE ( <mot> --- )
  ' >link dup l>name
  2dup prior @ hash
  begin 2dup @ =
    not while @
  repeat
  >r @ r> !
  remote @ 2+ hash
  2dup @ swap ! ! ;
```

SECRETAIRE Du 27.06.89 A 09h05
CHANGEMENT D'ADRESSE FIG R.F.A.:
FORTH-Gesellschaft e.V.
Postfach 11 10
D-8044 UNTERSCHLEISSHEIM
Tel: 19-49 89-317 37 84 depuis la France
Mailbox: 19-49 89-725 96 25
300/1200 baud 8N1

SECRETAIRE Du 28.06.89 A 15h42
CA Y EST, MA SUPER APPLICATION DE TELES AISIE PAR MINITEL
80 COLONNES CARBURE AU QUART DE TOUR:
- DEFINITION DE CHAMPS DE SAISIE PAR TYPE DE DONNEES
(DATE, NOMBRES, CHAINES MAJUSCULES ET CHAINES
ORDINAIRES)
- GESTION DE L'ECRAN MINITEL 1B EN MODE PLEIN ECRAN
- LORS DE SAISIES EN SERIE, POSSIBILITE DE RETOUR AU
CHAMP PRECEDENT COMME AVEC LE READ DE dBASE
- TRANSFERT DES DONNEES DANS UN FICHIER dBASE
DEMO SUR VOTRE MINITEL 1B EN APPELANT AUX HEURES DE
BUREAU AU 1-49856315 LES LUNDI, MARDI, JEUDI ET
VENDREDI.

CETTE APPLICATION DE TELES AISIE DE DONNEES EST
OPERATIONNELLE DANS LE CADRE DES ACTIVITES 'GENIE CIVIL,
ETUDE DES SOLS' A FRANCE TELECOMM.

LE CHOIX DE FORTH A ETE BENEFIQUE SUR LE PLAN RAPIDITE
DE DEVELOPPEMENT. EN EFFET, UNE EQUIPE C ESSAIE
VAINEMENT DE REFAIRE LE PROGRAMME, ET MALGRE MON AIDE,
PATAUGE JOYEUSEMENT...

JE 01 N°52 PAILLET 1989

UNE VERSION JEDI DES PRINCIPALES ROUTINES SERA REALISEE ET MISE A DISPOSITION DES ADHERENTS JEDI DANS UN PROCHAIN MODULE ET EN TELECHARGEMENT. A+

FORTH7 Du 02.07.89 A 19h30
A QUOI PEUT BIEN SERVIR CE MOT ?

```
: STRUCTURE
  CREATE ,
  DOES> @ CREATE DUP , ALLOT
  DOES> 2+ ;
```

Deux CREATE et deux DOES> dans un meme mot? Serait-ce:

- Un mot definissant des mots qui se definissent eux-memes
 - Un mot definissant des matrices a deux dimensions
 - Un definisseur de mots de definitions
 - Un mot de definitions a double sens
- JEDI! POUR LA PREMIERE BONNE REPONSE

SECRETAIRE Du 03.07.89 A 09h24
DEUX CREATE ET DEUX DOES, tres drôle.. Et plus surprenant est l'execution de STRUCTURE quand on fait:
DEBUG STRUCTURE
10 STRUCTURE MACHIN
MACHIN CHOSE

Pourtant, quand on debugue seulement STRUCTURE, CHOSE ne devrait pas se mettre en execution de la partie DOES> de MACHIN, logiquement...

ENCORE PLUS CURIEUX EST LE PROBLEME suivant:

```
: MANIPULATION
  CREATE ,
  DOES> @
  DOES> @ 2+ ;
```

Comment va fonctionner le mot defini par n MANIPULATION <mot>? Genetique, mon cher Watson, genetique...

ET MAINTENANT SANS CREATE, UNIQUEMENT AVEC DOES>, voici la bete definition:

```
: BEBETE ( n ---)
  1+
  DOES>
  2+ ;
```

Si vous vous tapez la tete sur le mur, mettez au moins un casque...

FLIPO Du 03.07.89 A 12h19
Quelqu'un peut-il m'indiquer ou je peux trouver le format des fichiers texte :

- DCA (Document Conversion Architecture) d'IBM
- RTF (Rich Text Format) de Microsoft

Je souhaite realiser un programme de conversion des fichiers textes ecrits sur ATARI avec "Le Redacteur" par ex. en fichiers utilisables par un traitem. de texte sur PC ou MAC tel que WORD, ceci en gardant au maximum les attributs du texte initial (soulignement, italique tabulations etc..), le transfert en code ASCII ne me suffit donc pas ! Ce type de conversion passe par un codage intermediaire en DCA ou mieux RTF.

SECRETAIRE Du 03.07.89 A 17h16
REPONSE A FLIPO: le mieux est de demander ces renseignements directement a la Ste MICROSOFT:
MICROSOFT 12, ave Quebec
LES ULLIS ESSONNE
tel: 1-69.86.10.20 pour le support technique.

Pour mon probleme de codage des images de type .PCX, ils m'ont repondu avec diligence et efficacite. Encore tout mes remerciements a leurs techniciens...

FF32 Du 03.07.89 A 17h41
JE LANCE AUJOURD'HUI UN MAILING AVEC UNE SPECIFICATION DU SIMULATEUR F32, PRONONCER S32. SI VOUS ETES INTERESSES PAR UNE COPIE, DEPOSEZ VOTRE ADRESSE POSTALE A FF32.

FORTH7 Du 04.07.89 A 21h05
\\PAUSE\\

Vous ecrivez le big stuff du siecle en Turbo-Forth, disons un simulateur F32, et vous desirez placer dans votre mega-source quelques points d'arrets provisoires, histoire de controler au radar que la compilation de la bete ne franchit pas la ligne jaune.

Vous allez placer par-ci par-la le mot

\\PAUSE\\
dont voici le cahier des charges:
- a l'execution de ce mot l'interpretation du fichier en cours est suspendue, la main est rendue au clavier pour permettre d'entrer autant de commandes forth desirees:

words .S see <mot> echo on etc etc.

L'entree d'une commande vide par un simple <CR> fait reprendre aussitot la suite de la compilation du fichier. Une commande erronee arrete tout.

EDIT va dans le fichier au niveau de la derniere \\PAUSE\\.

' voulez pas essayer
de l'inventer vous-meme
avant de vous jeter sur <SUITE> ?

80 STRING PAUSE\$

```
: \\PAUSE\\ ( --- )
  STATE @ 0= IF
  WHERE
  BEGIN CR ." ? "
  PAUSE$ INPUT$
  PAUSE$ NIP WHILE
  PAUSE$ $EXECUTE

  REPEAT
  THEN ; IMMEDIATE
  \\ plus long a commenter qu'a ecrire !
```

FF32 Du 06.07.89 A 18h41
N'EXAGERONS TUT DE MEME PAS LE BIG STUFF DE S32...
A PART CA, A QUOI SERT FINALEMENT STRUCTURE? RIEN COMPRIS... UN IGNARE, MAIS CURIEUX.

FORTH7 Du 07.07.89 A 13h42
SUPERSTRUCTURES: STRUCTURE ne semble avoir inspire que notre secretaire, pris que vous etes dans la moiteur estivale...

Il s'agit d'un mot de definitions de mots de definitions, sous une forme ici trop simple se contentant de reserver pour chaque type un espace memoire en octets:

```
2 STRUCTURE VAR
4 STRUCTURE LONG
1024 STRUCTURE BLOC
```

VAR EUSE VAR ECH LONG DRINK
BLOC NOTE BLOC HAUS

STRUCTURE n'est pas qu'une curiosite: si jamais votre application a besoin de tant de 'structures' qu'un mot de ce genre (create does> create does>) est rentable, alors vous n'echapperez pas aux langages objets. Vous venez de toucher aux notions de classes et d'heritage: les objets ne sont pas loin ...

Le hic c'est qu'a moins de programmes monstrueux, ce genre de truc reste pure curiosite d'ecole si on y

reflechit bien. Le Forth-Objet existe mais on ne sait toujours pas quoi faire avec! Ceci est valable pour tous les LOO: tout le monde ne developpe pas un hypertalk.

SECRETAIRE Du 10.07.89 A 08h47

Puisque l'on en est aux series de devinettes, en voici une autre:

```
VARIABLE #RP
CODE -1!!
-2 # AX MOV #RP #) BX MOV
AX 0 [BX] ADD NEXT END-CODE
: ESSAI ( ---)
RPa #RP ! 10 0
DO CR I .
KEY ASCII R =
IF -1!! THEN
LOOP ;
```

Maintenant lancez ESSAI et avant que 9 ne s'affiche, appuyez sur la touche R. Marrant, non?

SECRETAIRE Du 11.07.89 A 12h45

ENCORE UN NOUVEAU SERVEUR FORTH EN Europe, FORTH-Mailbox de MUNICH (RFA), 1200/1200 ou 300/300 8 bauds, peut etre interroge en composant le:

19-49 89/7259625

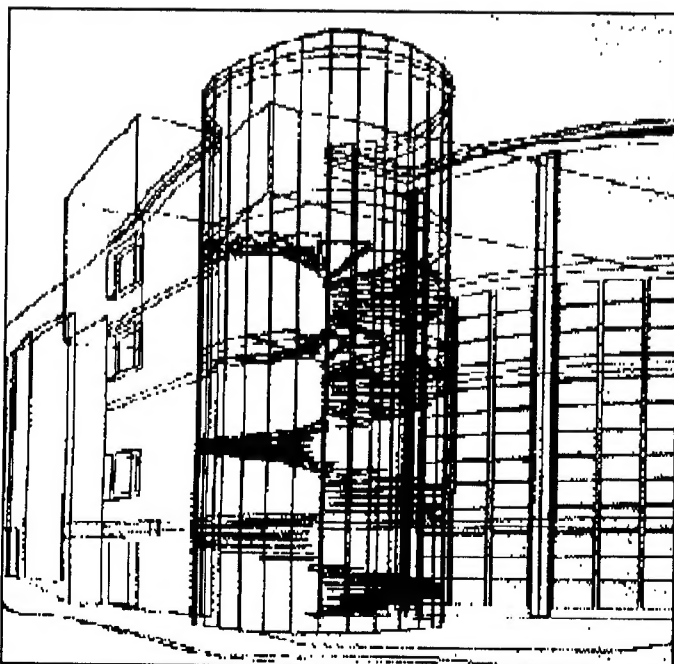
1 taxe toutes les 11 secondes en journee (contre 1 taxe/12 sec en interurbain...), tarif reduit la nuit. Nous ne l'avons pas encore essaye... Peut-etre des trucs interessants.

A ce jour, SAM*JEDI reste le serveur FORTH assumant le plus fort trafic parmi tous les serveurs FORTH connus a ce jour: EAST-COAST FORTH RBBS (USA), FORTH-Mailbox HAMBURG et FORTH-Mailbox MUNCHEN.

D'ailleurs, SAM*JEDI sera prochainement bilingue. Dans peu de temps, nous activerons le choix 8 du menu; a partir de ce moment, si votre pseudo est identifie comme connectant en langue anglaise, toutes les pages de presentation seront visualisees en anglais.

De prochains accords entre les reseaux BTX et TELETEL donneront acces aux connectants allemands transitant par BTX de disposer des services du 3615, donc de consulter SAM*JEDI. A plus long terme, une passerelle sera egalement cree entre le reseau PRESTEL (GB) et TELETEL.

Alors JEDI a decide de faire l'Europe du FORTH avant l'ouverture totale des frontieres.



Nous esperons que ces accords aboutissent rapidement. Potentiellement, JEDI pourrait deservir une communaute de programmeurs FORTH aussi importante, sinon plus que celle existant aux USA.

Les programmes crees et diffuses par nos adherents n'ont rien a envier a ceux diffuses aux USA dans FORTH DIMENSION. Ils sont meme meilleurs. Et l'idee d'une communaute de programmeurs FORTH specifiquement europeenne n'est pas utopique, car actuellement deja, des programmes ecrits POUR TURBO-FORTH sont diffuses par des etrangers dans des revues de programmation: VIERTE DIMENSION (RFA) et COMPUSER (NL) notamment, avant d'etre diffuses dans JEDI.

De notre cote, nous reservons une partie de la revue a des articles en VERSION ORIGINALE (anglais 80%, allemand 20%) afin de renforcer nos echanges et nos contacts avec les pratiquants FORTH non francophones.

FORTHFOUR Du 12.07.89 A 18h54

J'AI QQ CROSS ASSEMBLEER MOTOROLA SUR PC. DIFFUSE PAR MOTOROLA AU TRAVERS DU RESEAU DE DISTRIBUTION POUR POUSSER A LA CONSOMMATION. DONC:

AS1 POUR 6801
AS4 POUR 6804
AS5 POUR 6805
AS9 POUR 6809
AS11POUR 68HC11
ET X68000 POUR

ILS SONT SIMPLES MAIS POUR UNE 1ERE APPROCHE, C'EST SUFFISANT. ILS ONT UN PETIT MONOCHIP AVEC 3K DE ROM EN OTP, DRIVER DE LED POUR UNE TRENTAINE DE FRANCS, CA VOUS INSPIRE?

SECRETAIRE Du 12.07.89 A 19h03

REPONSE A FORTHFOUR:

ON EST INTERESSE PAR CROSS ASSEMBLER. POSSIBILITE D'APPLICATIONS DANS LE DOMAINE INDUSTRIEL ET AUTOMATISMES DE CES PRODUITS.

FORTH7 Du 12.07.89 A 22h43

TOUT MAIS PAS DE GOTO!

Puisqu'on est sur le meilleur serveur FORTH, m'en vais vous soumettre un probleme philosophique avant de partir pour reprendre quelques forth...

Vous savez que FORTH est un langage interprete et que nous disposons deja de directives interpretees de compilation. Pour les besoins d'une application hard-dependante, j'ai re-invente une horreur: GOTO (appelons-le ainsi comme en Basic ou mieux comme dans les fichiers batch du dos). Ca donne:

EXIST? COLOR ?\ GOTO PALETTE ou
KEY UPC ASCII 0 = ?\ GOTO OUIDA

Vous avez compris: il n'est vraiment pas difficile d'ajouter aux directives actuelles un mot immediat qui fait sauter l'interpreteur a un label dans le fichier source. D'aucuns trouveront ca assez pratique pour les compilations conditionnelles ou personnalisées.

Ouais, mais avez-vous deja relu un programme truffe de GOTO? L'enfer de la programmation-spaghetti! Tout mais plus ca! J'ai mis mon GOTO au frigo.

Je vous laisse juge et pars souffler un peu. Bye bye

FF32 Du 13.07.89 A 10h24

BONNES VACANCES A FORTH7; JE CROIS QU'IL EN A BIEN BESOIN... C'EST PAS LA BOUSCULADE POUR S32. QUI VEUT PLUS D'INFOS? SINON BON SOLEIL A TOUS.

EPICURE Du 14.07.89 A 21h05

GOTO OH POTO

Tout a fait d'accord avec FORTH7. Au frigo, qu'il les met? Je n'encombre pas le mien de choses aussi invouables.

FF32 Du 15.07.89 A 18h00
F32 CHIP DEVRAIT NORMALEMENT SE FAIRE COMME SUIV:
OCTOBRE 89 DEBUT ETUDE DU SILICIUM
TAPE-OUT EN JUIN 90
CHIPS PROTO EN SEPTEMBRE/OCT 90
ESSAI SUR CARTES PC(OU PC-AT) DEC 90
VALIDATION PRINTEMPS 91
..ON A LE TEMPS ?

PAS VRAIMENT. LE SIMULATEUR DOIT ETRE OPERATIONNEL AU PLUS
TARD DECEMBRE 89 POUR GUIDER LES METTEURS EN SILICIUM.

DE PLUS IL NOUS PERMET DE CHIADER L'ARCHITECTURE EN ESSAYANT
DIVERS CAS. ENFIN CE SERA L'OUTIL DE PRESENTATION DU CHIP
SANS ATTENDRE LA MATERIALISATION DE CELUI-CI. JEDI PUBLIERA
BIENTOT UN COMMENTAIRE DETAILLANT L'IMPLEMENTATION DE S32
POUR EN FAIRE CET OUTIL DE TRAVAIL AUSSI SOUPLE ET CLAIR
QUE POSSIBLE. UN DE SES BUTS EST DE NE PAS NECESSITER
L'EXPERTISE EN FORTH POUR SA MANIPULATION. A PLUS TARD...

LAMBERTPH Du 18.07.89 A 11h59
L'UN DE NOUS A-T-IL UN METAGENERATEUR SUR ATARI?

SECRETAIRE Du 18.07.89 A 16h40
REPONSE A PHLAMBERT: Si je ne me trompe pas, il y a un meta-
generateur FORTH avec VolksFORTH ATARI. En tout cas, les
sources sont dans les disquettes. Salutations.

FF32 Du 18.07.89 A 18h10
GOTO OR NOT GOTO? QUESTION PERTINENTE...EN 1970. MAINTENANT,
IL SERAIT TEMPS D'ELEVER LE DEBAT: LE MOT EST GRANULARITE.

LANGAGE A GRAIN FIN=COURSE VERS LA PREHISTOIRE.

GROS GRAIN(OBJETS)=PRODUCTIVITE PAR LA REUTILISATION DU
CODE. =AUSSI SECURITE DES APPLICATIONS.

QUE FAIT-ON AVEC LE FORTH POUR ATTAQUER CE PROBLEME? SI L'ON
SE CONTENTE DE DIRE ON PEUT TOUT FAIRE EN GRAIN FIN, LE
LANGAGE EST VU A RESTER ENTRE LES MAINS D'UNE POIGNEE
D'ARTISTE. MAIS AU FAIT, N'EST-CE PAS SON TENDRE AVANTAGE?
POUR MA PART, JE TROUVERAI CELA NAVRANT. AVIS PERSONNEL,
BIEN SUR.

SECRETAIRE Du 21.07.89 A 18h41
SI VOUS NE SAVEZ TOUJOURS PAS POURQUOI NOTRE ASSOCIATION
S'APPELLE JEDI, NE MANQUEZ PAS LE MARDI 25 JUILLET A 20H35
SUR A2, LE FILM DE G.LUCAS:
LA GUERRE DES ETOILES
(BIENTOT UN CULTE). LES MAQUETTES SONT ANIMEES PAR UN
ORDINATEUR PROGRAMME EN... FORTH

C'EST LA PREMIERE FOIS QUE L'ILLUSION EST DONNEE PAR LE
SUJET DE SE MOUVIR EN MEME TEMPS QUE LE DECOR. AVANT ON NE
FAISAIT QUE DES PLANS FIXES DANS LES SCENES TRUQUEES.

ALLEZ, BON FILM, ET: 'Que le FORTH soit avec vous!'

JACCOMARD Du 23.07.89 A 11h49
1 Grand merci a JLUC pour sa reponse a ma question de JEDI
#33. Rien trouve en 2 ans. La patience est recompensee.
2 JLUC demande (18/02) un turbo.exe. J'ai ecrit un F83.ASM
pour MASM v.5. Noyau reduit env. 14K, surtout pour charger
drctmt des fich. binaires. Editeur, debogueur, assembler
charges selon besoin en qq secondes. Interesse? Phone 98-73
70 81.

FF32 Du 23.07.89 A 19h49
C'EST VRAI QUE F32/S32 EST DIFFICILE A EXPLIQUER. BIENOT
PLUS D'EXPLIC+SCHEMA BLOC DANS JEDI.
EN ATTENDANT J'IRAI VOIR PERSONNELLEMENT QUELQUES-UNS POUR
FAIRE PASSER LE TOPO DE VIVE VOIX. ILS M'AIDERONT PEUT-ETRE
A CLARIFIER LE BEURRE. DESEPOIR, JAMAIS!

SECRETAIRE Du 24.07.89 A 11h05
L'elaboration d'un simulateur F32 est une entreprise assez
hardie:

- modifications profondes du meta-generateur de

TURBO-Forth.

- creation d'un source F32 compilable par ce meme
meta-generateur
- definition du simulateur a partir des
caracteristiques precedentes.

Sur le premier point, il faut nottament compiler toutes
les valeurs literales en format MOTOROLA. Pour
precision, en format MOTOROLA, les paires d'octets sont
dans l'ordre 0h0l dans un mot 16 bits, alors qu'ils sont
dans l'ordre 0l0h en INTEL; la transformation se fait en
utilisant le mot FLIP:

TRUE CONSTANT MOTOROLA IMMEDIATE
: I>M (n1 --- n2)
[COMPILE] MOTOROLA IF FLIP THEN ;

et dans chaque definition utilisant le mot C, (version
ASSEMBLER) le faire precéder de I>M.

Il faut aussi generer un code sans en-tete et
relogeable. Le code FORTH F32 est un code 'CHAINAGE
DIRECT':

CODE truc ... NEXT END-CODE

en TF83, compile un en-tete 'CHAINAGE INDIRECT', c'est
a dire que le cfa indique l'adresse d'execution. En F32,
le meme en-tete a un cfa contenant l'instruction call
pointant sur la zone parametrique. Ensuite, la
compilation de truc dans une definition de type
deux-points compile le contenu du cfa de truc et non son
cfa!

: MACHIN TRUC ;

devient automatiquement en F32:

CODE machin call, truc NEXT END-CODE

NEXT de F32 est simplement un ret, il n'y a donc plus de
difference en F32 entre definition CODE et definition :
si ce n'est au niveau de la methode de compilation.

Le simulateur F32 sera veritablement capable d'executer
du code F32. En fonction des caracteristiques du
processeur F32 (non encore bouclees a ce jour), le code
F32 sera assez eloigne du code FORTH tel que vous le
pratiquez:

C! store 8 bits
W! store 16 bits
I! store 32 bits

WVARIABLE variable 16 bits
VARIABLE variable 32 bits

C@ fetch 8 bits
W@ fetch 16 bits
I@ fetch 32 bits

A la compilation, les valeurs literales sont de trois
grandeurs:

8 bits -128..+127
16 bits -32768..+32767
32 bits autres entiers > 16 bits signes

La distinction est faite automatiquement au moment de la
compilation. Toute valeur literale est precedee d'un
code:

clit, literal 8 bits
1 octet code, 1 octet valeur
wlit, literal 16 bits
1 octet code, 2 octets valeur
dlit, literal 32 bits
1 octet code, 4 octets valeur

Toute valeur 8, 16 ou 32 bits exprimee sous forme literale en memoire est convertie en valeur 32 bits signee dans le registre T. Le contenu precedent de T est transfere dans N le cas echeant. Le contenu de N passe dans la troisieme position de pile le cas echant. Exemple:

: SOMME 10 -3 + ;

devient apres compilation F32, en code:

```
CODE SOMME
clit, 10 clit, -3 +, ret,
END-CODE
```

Bien entendu, c'est la definition en langage evolue que vous aurez ecrite.

SECRETAIRE Du 24.07.89 A 13h34
LISTE DES SERVEURS FORTH EXISTANTS: protocole ASCII
300/1200/2400 bds, huit bits, sans parite, 1 bit stop aux
USA:

GENie 800-638-9636 (19-1) code FORTH
BIX (ByteNet) 800-227-2983 pour info, code BIX via TymeNet
puis jforth
COMPUSEVE 800-848-8990 pour info, FORTH ou CLM
UNIX BBS: 415-332-6106 (19-1)
415-753-5265 (19-1)
East Coast Board Forth: 703-442-8695
British Columbia Forth Board: 604-434-5886
Real Time Control Forth Board: 303-278-0364
(liste mise a jour en mars 1989)

LAMBERTPH Du 24.07.89 A 22h09
OU LES ATHENIENS S'ATTEIGNIRENT: MA QUESTION SUR LE
METAGENERATEUR SUR ATARI VIENT DE LA NECESSITE DE NETTOYER
UN FORTH PLEIN D'ASTUCES MAIS UN PEU BROUILLON DU SAN
LEANDRO COMPUTER CLUB. LA VERSION 32 BITS EST EN CODE
DIRECT, LES OCTETS DANS LE BON ORDRE PUISQU'ON EST SUR
MOTOROLA.. CELA NE RESSEMBLE-T-IL PAS AUX CARACTERISTIQUES
DE F32?

LE METAGENERATEUR LIVRE AVEC EST INCOMPLET, QUAND A LE
COMPLETER CELA DEMANDE UNE COMPREHENSION DU SUJET POUR LE
MOMENT HORS DE MES CONNAISSANCES. AMITIES
PS N'AYANT PAS DE GENERATEUR AVEC LE VOLKSFORTH, TOUTE IDEE
OU PROPOSITION SERA ETUDIEE.

SECRETAIRE Du 25.07.89 A 13h27
LE PROBLEME DE L'ORDRE DES OCTETS EST independant du
processeur utilise. En principe, on optimise le code par
rapport au micro-processeur utilise. Ca ne devient important
que lors d'un DUMP ou toute autre operation d'accès sur
l'espace memoire qui n'est pas definie. Le systeme de codage
INTEL est exploite sur les systemes equipes des muP
suivants: 8080 280 8088 8086 80286-386. Le muP HARRIS RTX
2000 exploite indifferemment les modes INTEL ou MOTOROLA. Le
systeme de codage memoire MOTOROLA est exploite par 6502
6800 6809 68000 etc... ainsi que AM2900 et les TRANSPUTER
INMOS T800.

En meta-generation, les mots sensibles a modifier sur META
de TURBO-Forth pour metagenere du code conforme au mode
MOTOROLA sont essentiellement !-T, -T.

En chainage direct, il faut toujours une structure de type:

nfa lfa cfa pfa si on compile avec en-tete
cfa pfa si on compile sans en-tete

En l'absence d'un cfa appropriée, l'interpreteur interne ne
peut se reporter aux parties execution des mots ayant un
type different des mots CODE. En conservant une structure
homogene, on evite de gerer trois piles:

- pile retour systeme
- pile retour FORTH
- pile donnees FORTH

et on fait partager les deux premieres piles par FORTH.

Sur certains muP, les registres de pointeurs de pile
sont modifiables a volonte, permettant de basculer d'une
pile a l'autre, ce qui n'est pas le cas des
systemes a base INTEL.

Sur INTEL, les seules operations incrementant ou
decrementant implicitement le pointeur de pile sont PUSH
et POP.

Sur MOTOROLA diverses instructions disposent
d'operations avec pre-decrementations simple et double
ou post-incrementation simple et double. La gestion de
piles auxiliaires en est facilitee.

En F32, il n'y a que deux modes d'adressage
envisageables:

- adressage immediat (clit, wlit,
dlit, rp@, sp@, rp!, sp!, ...)
- adressage inherent (dup, drop,
swap, +, ...)

Un acces en mode adressage indirect est defini par une
suite de codes:

dlit, xxxxxx @, @,
xxxxxx etant une adresse servant d'index

Un papier detailant ces mecanismes sera diffuse aux
participants du projet F32.

FLIPO Du 25.07.89 A 19h11

FORTH-32BITS SUR ATARI: ENFIN !!!

BRADLEY FORTHWARE P.O. Box 4444

Mountain View, CA 94040

diffuse en "shareware" un forth 32-bits pour ATARI, dont
des variantes sont disponibles pour MAC (512ko au moins
a IIX) pour stations SUN, systemes 68020 sousUNIX... Les
fichiers sont ASCII, comme TURBO FORTH, pas de blocs.

Je commence a peine a le prendre en main; le produit me
semble tres interessant. Fin de la limitation a 64k,
quel pied! Je suis autorise a vous donner gratuitement
pour essai le "Working disk". Pour le recevoir m'envoyer
une disquette Sfet une enveloppe timbree a 3,70 Frs,
portant votre adresse. La mienne:

Daniel FLIPO 1 bis rue St-Jacques 59800 LILLE.

Si le produit vous plait vous pourrez le commander (avec
doc de 200 pages tres complete) pour 50 US\$ (# 350 Frs).
Le source du noyau + metacompilateur coute 50\$ en
plus, licence develt incluse. L'interface GEM (AES+VDI,
macros pour fenetres etc..) coute 25\$.

La version MAC coute aussi 50\$, la version SUN 150\$
(source compris). Paiement: Cheque FRANCAIS en \$, ou
virement bancaire, NO CREDIT CARD. Je trouve que le
travail vaut largement ces prix, je suis POUR le
developpement du "shareware", je ne cederai donc RIEN
D'AUTRE que le "working disk". SORRY!! Si vous commandez
mentionnez JEDI (il faut bien se faire connaitre
la-bas!).

L'auteur de ce FORTH cherche aussi a faire une
metacompilation automatique des seuls mots utiles d'une
application, mais a abandonne... Michel ZUPAN
releve-ra-t-il le gant lorsqu'il sera bien repose a son
retour de vacances??? ALLEZ JEDI!

FF32 Du 26.07.89 A 19h41

FAUT-IL CONNAITRE LE TEXTE AVANT D'ECRIRE UN EDITEUR DE
TEXTE? FAUT-IL CONNAITRE LA STRUCTURE DU F32 AVANT
D'ECRIRE LE MONITEUR DE SIMU? JE M'INTERROGE...

ET SI JE DONNE UNE STRUCTURE, ET QU'IL FAUT LA FAIRE
EVOLUER; QUEL POURCENTAGE FAUDRA-T'IL REECRIRE? 99%?
NON... HELP ME JEDI! JE SENS UN GRAND BOULEVERSEMENT DU
FORTH...

SECRETAIRE Du 27.07.89 A 11h07

REPONSE A FF32: un outil est toujours adapte a la
matiere a traiter: meche a bois, meche a beton, meche

JEDI N° 12 - JUILLET 1989

acier, meche de 8, 10, 12,... perceuse a main, electrique, autonome, a percussion...

Il en sera de meme pour le simulateur F32; comme nous n'avons pas l'habitude de travailler en comite avec elaboration d'un cahier des charges (dixit METHODE MERISE...) a respecter, car nous ne savons pas ou nous mettons les pieds, la solution dite "Essais/erreurs" suivie du cycle "Correction/amelioration" me semble la plus efficace.

Rappel a FF32 et information pour les autres: c'est la premiere fois dans l'histoire de la micro-informatique que tous les outils de developpement d'un processeur soient diffuses avant sa fabrication; de meme diffuserons-nous toutes les caracteristiques de F32.

Cette demarche est un pari:

- pari sur l'outil telematique; vous participez en donnant votre avis; vous telechargez l'outil de developpement.
- pari sur l'interactivite; fini le petit comite; on

rassemble les competences les plus disparates (esperons que ca ne devienne pas la tour de BABEL...)

Tres bientot je met en telechargement l'assembleur F32 modifie. A+.

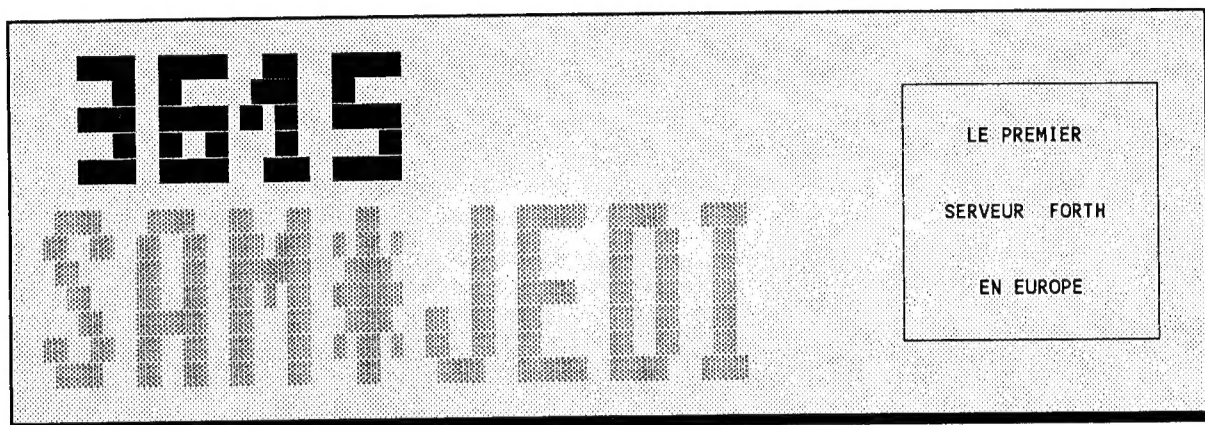
FF32

Du 27.07.89 A 18h14

PAS QUE JE RESISTE A EMETTRE DES PLANS MAIS JE TIENS A SIGNALER QU'IL Y A UN QUIPROQUO, A MON AVIS, SUR CE QUE L'ON ENTEND PAR SIMULATEUR F32. IL ME SEMBLE QUE MARC PARLE D'UNE MACHINE VIRTUELLE 32 BITS EXECUTANT UN ASSEMBLEUR AD-HOC.

JE DECRIS UN MONITEUR QUI, AU NIVEAU LE PLUS SIMPLE, ACTIVE CE TYPE DE COMPORTEMENT. MAIS C'EST ACCESSOIRE POUR LE MOMENT, ET C'EST LA STRUCTURE DUDIT MONITEUR QUE JE DECRIVAIS DANS MON MAIL DU DEBUT DU MOIS.

L'AVIS DES DESTINATAIRES ME SERAIT PRECIEUX, VOIR QUI A SAISI QUOI. FEEDBACK, PLEASE! QUI VEUT + DE PRECISIONS? DEMANDEZ COMMUNIQUEZ...



Bienvenue sur SAM * JEDI

- 1 Vous ouvrir une BAL-JEI
- 2 Ecrire un message
- 3 Lire votre bal
- 4 Lire le forum
- 5 Consulter l'annuaire
- 6 Relire anciens msgs Forum
- 7 TELECHARGEMENT
- 8 Sélection type terminal

Tapez votre choix N . + ENVOI
Utilisation de SAM*JEDI GUIDE

Welcome to SAM * JEDI

- 1 You open a "post box"
- 2 Write a message
- 3 Read your post
- 4 Read the forum
- 5 Consult the directory
- 6 Reread past Forum msgs
- 7 TELECHARGING
- 8 Selection of terminal type

Type your choice N . + SEND
Utilisation of SAM*JEDI GUIDE

